Atte Väyrynen

JÄTEVEDENPUHDISTAMON VALVOMOSOVELLUS

JÄTEVEDENPUHDISTAMON VALVOMOSOVELLUS

Atte Väyrynen
Opinnäytetyö
Kevät 2018
Sähkö- ja automaatiotekniikan
tutkinto-ohjelma
Oulun ammattikorkeakoulu

TIIVISTELMÄ

Oulun ammattikorkeakoulu
Sähkö- ja automaatiotekniikan tutkinto-ohjelma, automaatiotekniikka

---

Tekijä: Atte Väyrynen
Opinnäytetyön nimi: Jätevedenpuhdistamon valvomosovellus
Työn ohjaajat: Timo Heikkinen (OAMK) ja Sami Österman (Protacon Technologies Oy)
Työn valmistumislukukausi ja -vuosi: Kevät 2018 Sivumäärä: 61 + 7 liite

---

Opinnäytetyön tavoite oli suunnitella ja tehdä uusi valvomosovellus Äänekoskelle Teräväniemen jätevedenpuhdistamolle Äänekosken Energia Oy:n määritysten mukaisesti. Työ suoritettiin Protacon Technologies Oy:lle Oulussa.

Valvomosovelluksen avulla jätevedenpuhdistamoa voidaan tarkkailla sekä ohjata. Sovellus tehtiin vanhaan projektiin uutena osiona, jonka tuli pitää sisällään jätevedenpuhdistamon osaprosessit välipumppaamo, ilmastus ja jälkiselkeytys. Käyttöliittymään piti tehdä muun muassa uudet tiedonsiirtopisteet (tagit), ikkunat, objektit, hälytykset ja käyrästöt.

Valvomon käyttöliittymä tehtiin käyttäen Ignition SCADA -ohjelmaa, joka on tarkoitettu automatisoitujen prosessien valvomosovellusten tekemiseen. Ignition-ohjelmaan on ollut helppo tutustua käyttämällä hyväksi sen valmistajan nettisivuilla olevia opastusvideoita, jotka kertovat Ignitionin ominaisuuksista ja toiminnoista.

Opinnäytetyöni raportti pitää sisällään kuvauksen Ignition-ohjelmasta, jossa käydään läpi sen yleiset pääperiaatteet ja toiminnot. Tämän lisäksi raportissa on teoriaosuus siitä, minkälainen on visuaalisesti hyvä käyttöliittymä, jossa tiedon perustana käytetään hyväksi kirjallisia lähteitä, kuten esimerkiksi kirjoja ja oppaita. Viimeisenä osiona raportissa valvomosovelluksen suunnittelu, jossa kerrotaan päävaiheet siitä, miten valvomosovellus on rakennettu.

Asiasanat: valvomo, Ignition SCADA, käyttöliittymä, jätevedenpuhdistamo

ALKULAUSE

Haluan kiittää Protacon Technologies Oy:tä mahdollisuudesta suorittaa opinnäytetyö heille. Kiitos erityisesti työni ohjaajille Site Manager Sami Östermanille ja lehtori Timo Heikkiselle kaikesta tuesta ja avusta. Kiitos myös koko Oulun Protaconin henkilökunnalle kaikista neuvoista.

Kiitos tukijoukoille kotona kaikesta tuesta neljän vuoden opiskelun aikana.

Oulussa 5.2.2018

Atte Väyrynen

SANASTO

CSV	Comma-separated values on taulukkorakenteinen tiedosto-muoto, jossa solut erotetaan pilkulla.
FAT-testi	Factory Acceptance Test; toiminnallinen tarkastus
Ignition Gateway	Yhdyskäytävä
Java	Oliopohjainen ohjelmointikieli
LabVIEW	Ohjelmistoympäristö
MySQL	Tietokantaohjelmisto
OPC	Avoin tiedonsiirronstandardi
Oracle	Tietokantaohjelmisto
PLC	Programmable Logic Controller on pieni tietokone, jota käytetään automaatioprosessien ohjauksessa.
Python	Tulkattava ohjelmointikieli
SCADA	Supervisory control and data acquisition; Valvomo
SVG-grafiikka	Scalable Vector Graphics; skaalautuva vektorigrafiikka

VPN	Virtual Private Network; virtuaalinen erillisverkko, jonka avulla voidaan yhdistää fyysisesti erillään olevia aliverkkoja
XML	Extensible Markup Language; rakenteellinen kuvauskieli, jota käytetään esimerkiksi tiedostojen tallennusformaattina

SISÄLLYS

1 JOHDANTO

Työn tavoitteena oli toteuttaa valvomosovellus Äänekosken Teräväniemen jätevedenpuhdistamolle. Työn tilaajana toimi Äänekosken Energia Oy ja toimittajana Protacon Technologies Oy. Valvomosovellus tehtiin Protacon Technologies Oy:n tiloissa Oulussa.

Työhön kuului valvomosovelluksen konfigurointi, joka sisälsi muun muassa prosessinäyttöjen piirron, tiedonsiirtopisteiden (tagien) luonnin logiikkaohjelman toimittajan määritysten mukaisesti, hälytysten käsittelyn ja trendit. Tätä opinnäytetyötä varten olen tehnyt koko urakasta ensimmäisen vaiheen, joka pitää sisällään osaprosessit nimeltään välipumppaamo, ilmastus ja jälkiselkeytys. Olen siis konfiguroinut valvomosovelluksen vain näihin osaprosesseihin.

Äänekoskelle oli jo aiemmin toimitettu valvomosovellus, mutta nyt uuden jätevedenpuhdistamon valmistuessa täytyi vanhaan valvomoprojektiin lisätä uusi osio uudistetulle puhdistamolle. Minun tekemäni valvomosovellus on siis luotu vanhan projektin päälle laajennuksena.

Vanhasta sovelluksesta on ollut minulle paljon hyötyä, koska sieltä olen voinut ottaa mallia sovelluksen tekemiseen liittyvistä käytänteistä. Työni olikin hyvin pitkälti vanhaan ohjelmaan tutustumista, jonka tuloksena olen voinut käyttää oppimaani myös hyväksi omassa tekemisessäni. Joitakin valvomon toimintoja oli tehty jo valmiiksi, joten niihin ei tarvinnut muutoksia. Tässä opinnäytetyön raportissa pyrin käymään läpi tekemiäni toimintoja ja vanhasta kopioimiani objekteja, joita olen muokannut sovellukseeni sopivaksi.

2 TERÄVÄNIEMEN JÄTEVEDENPUHDISTAMO

Teräväniemen jätevedenpuhdistamo sijaitsee Äänekoskella ja sen omistaa Äänekosken Energia Oy. Äänekosken Energia Oy pitää huolen Äänekosken, Suolahden, Sumiaisten ja Konginkankaan vesihuollosta. Yritys rakentaa ja ylläpitää vesi- ja viemäriverkostoja ja huolehtii veden puhdistuksesta sekä juomaveden laadusta alueillaan. Äänekosken Energia Oy tuottaa myös sähköä ja lämpöä Äänekosken kaupungille ja sen lähiseuduille. (Äänekosken Energia Oy 2018, viitattu 25.2.2018.)

70-luvulla rakennettuun Teräväniemen jätevedenpuhdistamoon johdatetaan vesiä Äänekoskelta sekä Valion tehtaalta. Lähes kaikki jätevedenpuhdistamon prosessiyksiköt saneerataan toukokuuhun 2018 mennessä. Nykyään laitosta ohjataan pitkälti käsin ja sen automaatioaste on pieni. Saneerauksen tavoitteena on automatisoida prosessinohjaus ja tuoda puhdistamolle nykyaikainen paikallisvalvomo, josta prosessia voidaan ohjata sekä seurata. Saneerauksen myötä jätevedenkäsittelyn käyttökustannukset vähenevät, koska puhdistusteho paranee. (Äänekosken kaupunkisanomat 2017, viitattu 25.2.2018.)

Saneerauksessa jätevedenpuhdistukseen rakennetaan kokonaan uusia prosessiyksiköitä, kuten esimerkiksi uusi typenpoisto- ja jälkikäsittelylaitos. Myös allastilavuuksia kasvatetaan. Näillä uudistuksilla saavutetaan entistä tiukemmat ympäristölupaehdot, jotka astuvat voimaan vuoden 2018 alusta. (Sama.)

Saneerauksen ensimmäisessä vaiheessa valvomosovelluksen tuli sisältää osaprosessit nimeltään välipumppaamo, ilmastus ja jälkiselkeytys. Välipumppaamon tilavuus on noin 50 m^3 ja siinä esiselkeytetty vesi nostetaan biologiseen käsittelyyn, jonka maksimi virtaus on noin 700 m^3/h. Poikkeustilanteissa voidaan välipumppaamosta kaikki vesi viedä luukun kautta jälkikäsittelyyn. Pumppaamoa voidaan ohjata joko pintarajaohjauksella tai vakiopintaohjauksella. Pintarajaohjauksessa välipumppaamoaltaan pinta voi vaihdella asetetuissa rajoissa, kun taas vakiopintaohjauksessa altaalle asetetaan tavoitepinta. (Koskinen 2017, 18.)

Ilmastus on biologisen käsittelyn ensimmäinen vaihe. Ilmastuksessa jäteveteen sekoitetaan biomassaa eli aktiivilietettä, joka sisältää mikrobeja. Mikrobien tarkoitus on päästä kosketukseen vedessä olevia epäpuhtauksien kanssa, jotta ne saadaan muutettua mikrobeiksi eli hiilidioksidiksi ja vedeksi. Ilmastusaltaihin syötettävä ilma varmistaa sen, että mikrobeilla on tarpeeksi happea. Lopuksi altaassa syntyneet mikrobiflokit laskeutuvat altaan pohjalle, ja altaan päälle jää puhdas vesi, joka johdatetaan jälkiselkeytykseen. Teräväniemen jätevedenpuhdistamossa on kolme ilmastusallasta, joista ylijäämälietteet kootaan yhteen. Altaissa on sekoittimia, jotta mikrobit pääsevät kosketukseen epäpuhtauksien kanssa. Sekoituksesta johtuva veden liike estää myös syntyvien flokkien laskeutumista ilmastusaltaan pohjalle, kun veden puhdistus on vielä kesken. (Kabata 2018, viitattu 25.2.2018)

Jälkiselkeytys muodostaa yhdessä ilmastuksen kanssa biologisen käsittelyn. Selkeytyksen tarkoituksena on erottaa ilmastuksessa ylläpidettävä biolietekanta vedestä ja pumpata liete takaisin ilmastukseen. Vedestä liete saadaan erotettua painovoiman avulla jälkiselkeytysaltaan pohjalle, ja altaan pinnalle jäävä puhdas vesi kuljetetaan seuraavaan käsittelyyn. (Koskinen 2017, 29.)

3 IGNITION SCADA -VALVOMOSOVELLUS

Tässä luvussa käydään läpi Ignitionin perusteet ja pääperiaatteet sekä sen ominaisuuksia. Pyrin kertomaan suurimman osan niistä ominaisuuksista, mitä olen valvomosovelluksen suunnittelussa itse käyttänyt.

3.1 Perusteet

Ignition on SCADA-sovellus ja sen on tehnyt yritys nimeltä Inductive Automation. SCADA (Supervisory Control And Data Acquisition) tarkoittaa sovellusta tai laitteistoa, jolla prosessia ohjataan ja siitä kerätään dataa. Käyttöliittymä on osa SCADA:a ja sitä voidaan käyttää esimerkiksi tehtaiden valvomoissa SCADA-toimintaan.

Ignition on tarkoitettu automatisoitujen prosessien valvomoiden käyttöliittymien tekemiseen. Ignitionia ostaessa asiakas saa käyttöönsä Ignition-palvelimen lisenssin. Tähän palvelimeen on mahdollista yhdistää rajaton määrä asiakkaita ja luoda myös rajaton määrä tageja. Palvelimeen yhdistettävillä asiakkailla tarkoitetaan koneita eli esimerkiksi tietokoneita, tabletteja tai puhelimia, jotka ovat yhteydessä valvomoon. Asiakkaat pystyvät näkemään valvomosovelluksen ajotilassa, joka tarkoittaa sitä, että asiakkaat näkevät prosessissa vallitsevat olosuhteet reaaliajassa.

Ignition-palvelimelle voi yhdistää rajattoman määrän laitteita eli esimerkiksi PLC:itä (Programmable Logic Controller, ohjelmoitava logiikka). PLC sisältää tulo- ja lähtöportteja. Tuloihin voidaan kytkeä esimerkiksi antureita, joilta saadaan tietoa prosessissa vallitsevista olosuhteista. Lähtöjen kautta PLC ohjaa prosessia siihen ohjelmoidun ohjelman perusteella. Logiikkaohjelmassa voidaan käyttää hyväksi tuloarvoja, joiden ehdoilla lähtöjä voidaan kontrolloida. Logiikan tulot ja lähdöt ovat joko digitaalisia tai analogisia. Digitaaliset viestit ovat binäärisiä, joka tarkoittaa sitä, että viestit voivat olla arvoltaan vain nolla tai yksi. Yleisimpiä analogiviestejä ovat virta- ja jänniteviestit. Virtaviestit ovat yleensä 4–20 mA tai 0–20

mA ja jänniteviesti on yleensä 0–10 V. PLC:n lähtöjä voidaan käyttää esimerkiksi moottorin käynnistämiseen tai venttiilin asennon määrittämiseen. PLC:n avulla on helppo korvata aiemmin prosessinohjauksessa paljon käytettyjä releitä ja ajastimia.

3.2 Pääperiaatteet

Ignition-valvomosovellus on Javalla ohjelmoitu verkkopohjainen sovellus. Verkkopohjaisuus tarkoittaa sitä, että projekteja luodaan sekä konfiguroidaan selaimessa. Selaimesta sovelluksen käyttäjä voi avata Ignition Gatewayn, jonka kautta valvomosovelluksen voi avata. Gatewayltä valvomosovelluksen suunnittelija pystyy konfiguroimaan esimerkiksi valvomoprojektin hälytykset, historiatiedonkeruun ja tietokantayhteydet. Gatewayltä pystyy avaamaan kehitystyökalun, jolla valvomosovellusta voi rakentaa. Kaikki kehitystyökalussa tehdyt muutokset tallentuvat Gatewaylle, josta sinne yhdistetyt asiakkaat eli esimerkiksi valvomon tietokone näkevät tallennetut muutokset heti. Kuvassa 1 näkyy esimerkki Ignitionin arkkitehtuurista. (Inductive University 2018q, viitattu 25.1.2018.)

KUVA 1. Ignition valvomo-ohjelmiston arkkitehtuuri (Inductive University. 2018q)

Ignition käyttää logiikan kanssa kommunikointiin OPC-protokollaa. Ohjelma sisältää oman OPC-palvelimen, joka sisältää ajurit yleisimmille logiikoille. Ignition on myös mahdollista yhdistää yleisimpiin tietokantoihin kuten esimerkiksi MySQL:ään. Valvomon yhdistäminen logiikkaan ja tietokantaan tapahtuu Ignition Gatewayltä. (Sama.)

Ignitionissa on erilaisia moduuleita eli lisäosia. Ne tuovat sovellukseen lisäominaisuuksia, joka tarkoittaa sitä, että mitä enemmän moduuleita Ignitionissa on, niin sitä enemmän ominaisuuksia ja työkaluja on käytettävissä valvomosovelluksia tehtäessä. “Ignition on kuin puhelin. Puhelimessa asennetaan applikaatioita ja Ignitionissa moduuleita” (Inductive University 2018a, viitattu 25.1.2018). Moduulit mahdollistavat sen, että sovelluksen käyttäjä voi valita haluamansa lisäominaisuudet tarpeidensa mukaan. (Inductive University 2018a, viitattu 25.1.2018.)

3.3 Ominaisuudet ja työkalut

Ignition sisältää monia ominaisuuksia ja työkaluja sekä kehitystyökalussa että Gatewayllä. Kehitystyökalussa keskitytään enemmän itse valvomosovelluksen luomiseen, kuten ikkunoiden piirtoon sekä tagien luontiin. Gatewayllä taas muokataan projektin pääasetukset ja ominaisuudet, jotka heijastuvat myös kehitystyökalun puolelle. Kun Gateway on konfiguroitu hyvin, on helpompi alkaa työstämään valvomosovellusta kehitystyökalussa, koska niin moni asetus vaikuttaa kehitystyökaluun Gatewayn puolelta.

3.3.1 Ikkunat

Ignitionilla tehty valvomosovellus on projekti, joka koostuu kokoelmasta ikkunoita, joiden välillä operaattori liikkuu. Ignitionissa ikkunoita on kolmea eri tyyppistä, jotka ovat pääikkuna, ponnahdusikkuna ja kiinnitysikkuna. Jokaiselle ikkunalle annetaan nimi ja ne jaotellaan erillisiin kansioihin, jotta projektin kasvaessa niitä on helppo etsiä. Ikkunoiden kansiopolkuun viitataan ohjelmaa tehtäessä, jonka

takia kansiopolun suunnitteluun ja rakentamiseen kannattaa käyttää aikaa. (Inductive University 2018c, viitattu 25.1.2018.)

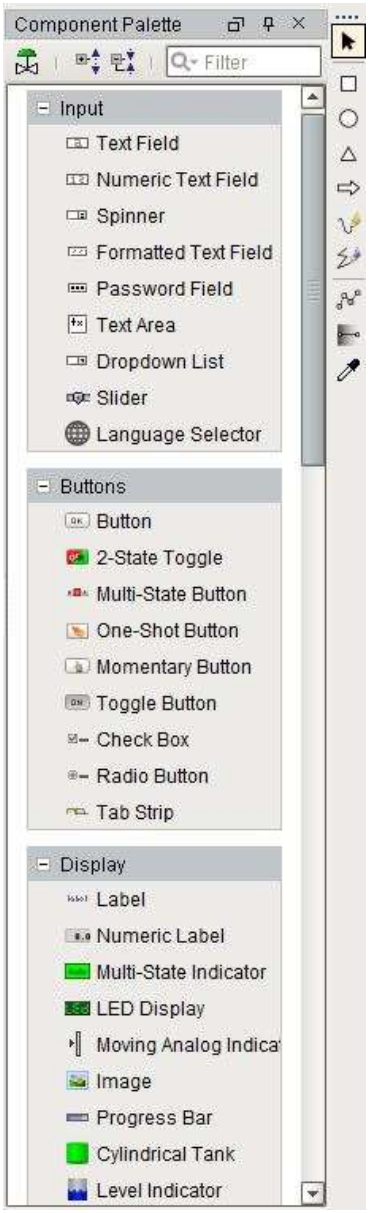
Ponnahdusikkuna on yleensä kelluva eli se aukeaa alla olevan pääikkunan päälle. Sen kokoa ja sijaintia voi muuttaa valvomon ollessa ajotilassa. Ponnahdusikkunoita voidaan käyttää esimerkiksi moottoreiden operointi-ikkunana. Kiinnitysikkuna taas yleensä pidetään vakiokokoisena sekä paikallaan, vaikka siirryttäisiin ikkunasta toiseen. Ignitionissa niitä voi sijoittaa joko ikkunan ylä- tai alareunaan tai jompaan kumpaan ikkunan sivureunaan. Niihin voidaan esimerkiksi sijoittaa lista (kuva 2), josta klikkaamalla operaattori pääsee navigoimaan ikkunalta toiselle. (Inductive University 2018t, viitattu 25.1.2018.)

KUVA 2. Kiinnitysikkunaan tehty navigointipalkki

Erityyppiset ikkunat poikkeavat toisistaan. Ikkunoilla on sen tyypistä riippuvia oletusasetuksia, joita pystytään muokkaamaan ikkunan luomisen jälkeen. Esimerkiksi pääikkuna tulee sovellukseen niin, että se ei sisällä reunoja, kun taas ponnahdusikkunaa tehtäessä tulevat reunat oletuksena mukaan ikkunaan. Tästä syystä käyttöliittymän suunnittelijan pitää tietää ikkunoita luodessa, onko ikkuna tarkoitettu pääikkunaksi, ponnahdusikkunaksi vai kiinnitysikkunaksi. (Inductive University 2018t, viitattu 25.1.2018.)

3.3.2 Piirtäminen ja Ignitionin omat komponentit sekä symbolit

Ignitionissa on omat piirtotyökalunsa. Kehitystyökalussa ne sijaitsevat kuvassa 3 olevan valikon oikeassa laidassa. Kuten kuvasta voi huomata, piirrettäviä kuvioita ovat muun muassa suorakulmio, ympyrä, kolmio ja viiva. Viiva-työkalulla ohjelmassa voi myös luoda omia kuvioita, kunhan viivan päät yhdistyvät eli kuviosta tulee sulkeutuva. Piirtotyökaluilla tehtyjen objektien muotoja sekä kokoa on mahdollista muokata monipuolisesti. (Inductive University 2018k, viitattu 25.1.2018.)



KUVA 3. Ignitionin piirto- ja komponenttivalikko

Kaikkia komponentteja ei tarvitse luoda manuaalisesti, vaan Ignitioniin sisältyy myös laaja valikoima erilaisia komponentteja aina painikkeista ja tekstikentistä taulukoihin ja kaavioihin asti. Näitä komponentteja käyttöliittymään voi raahata kuvassa 3 olevasta komponenttivalikosta, jossa näkyy vain osa saatavilla olevista komponenteista. Valvomoon komponenttivalikosta raahattuja komponentteja voi vapaasti liikutella ja muokata. (Inductive University 2018e, viitattu 25.1.2018.)
Ignitionin omilla komponenteilla on omat ominaisuutensa ja asetukset, joihin liitetään yleensä tagien arvoja. Komponenttien ominaisuuksia muokkaamalla saadaan esimerkiksi muutettua sen muotoa, kokoa, väriä tai arvoa. Jos kyseessä

olisi esimerkiksi numeerinen tekstikenttä, liittämällä sen oikeaan ominaisuuteen tagin arvo saadaan sen arvo näytettyä tai muutettua objektissa. Komponenttien tuominen valvomoikkunaan ei siis riitä siihen, että valvomo näyttäisi tai ohjaisi prosessia. Tätä varten komponenttien ominaisuudet tulee konfiguroida oikein, jotta valvomo toimisi halutulla tavalla. (Sama.)

Jokaista prosessilaitetta ei välttämättä tarvitse itse piirtää, koska Ignitionin mukana tulee myös paljon erilaisia symboleita. Nämä symbolit ovat yleisimpiä prosessilaitteita, mitä teollisuudessa käytetään, kuten esimerkiksi moottoreita, pumppuja, venttiilejä ja altaita. Symbolit sijaitsevat Ignitionin “Symbol Factoryssä” eli symbolitehtaassa, joka on Ignitioniin ostettu lisäominaisuus. (Inductive University 2018h, viitattu 25.1.2018.)

3.3.3 Tagit

Tagit ovat käyttöliittymän päivittyviä tietopisteitä ja ilman niitä valvomo olisi vain piirros vailla tietoa sekä ohjausta. Niitä siis käytetään esittämään valvomossa haluttuja arvoja ja kirjoituskohteena valvomosta päin. Jokaiselle PLC:ltä arvonsa saavalle tagille on määritetty jokin osoite ohjelmoitavalta logiikalta. Osoite on joko logiikan tulo- tai lähtöosoite. Tulo-osoitteista tagi saa arvonsa ja lähtöosoitteen sisältävät tagit ovat kirjoitettavissa valvomosta. Tagit toimivat siis osana logiikkaohjelmaa, ja kun niihin kirjoitetaan arvoja, ne ohjaavat prosessia logiikkaan ohjelmoidun ohjelman määrittelemällä tavalla. Tagi voi olla myös valvomosovelluksen sisäinenmuuttuja, jota voidaan käyttää hyväksi sovelluksen suunnittelussa. Esimerkki Ignitionin sisäisestämuuttujasta on ohjelman mukana vakiona tuleva tagi nimeltä ”CurrentDateTime”, joka näyttää päivämäärän ja kellonajan aikavyöhykkeelläsi.

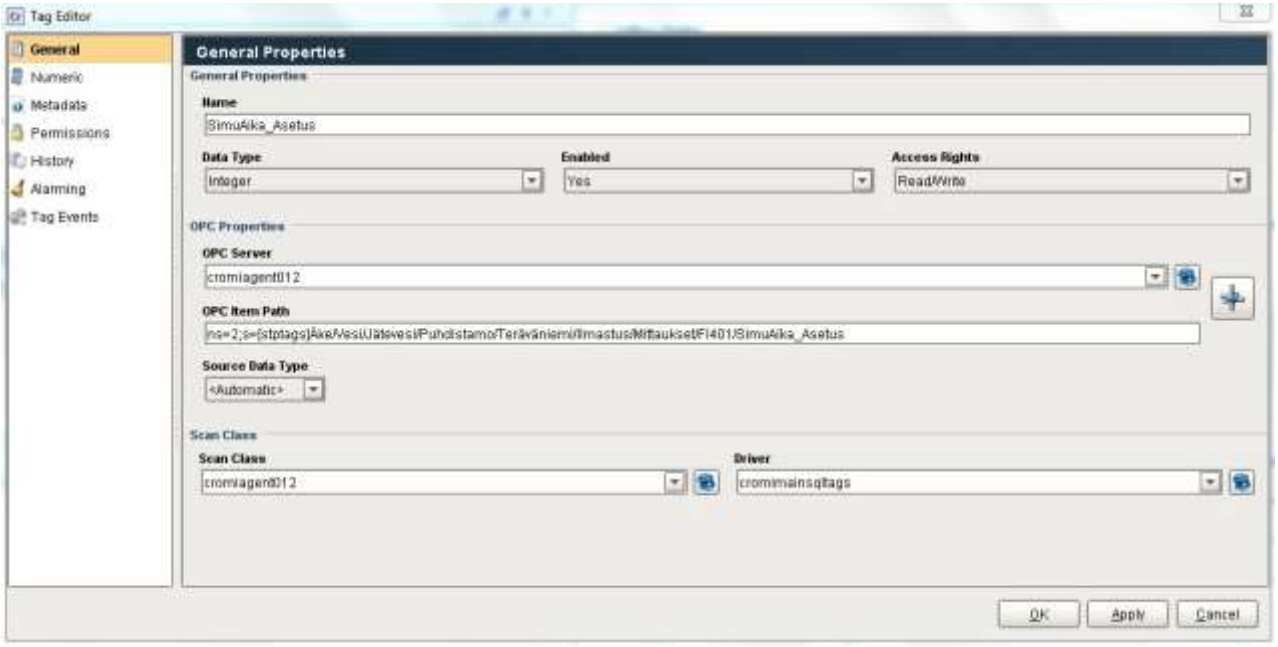
Ignitionissa tagit määritetään Gatewaylle. Sinne tageille luodaan tagikantoja, jotka näkyvät kaikille eri projekteille, jotka ovat yhdistettynä samaan palvelimeen. Tagit eivät siis ole pelkästään projektikohtaisia, vaan muistakin projekteista voi

nähdä muiden projektien tagit. Yleensä on kuitenkin tapana tehdä jokaiselle projektille oma tagikantansa, joka valitaan projektin oletuskannaksi. (Inductive University 2018r, viitattu 25.1.2018.)

Ignitionissa tageja on viisi erilaista, joita Ignitionissa voi käyttää hyväksi eri tilanteissa. Nämä eri tyypit ovat OPC-tagi, Expression-tagi, Query-tagi, Memory-tagi ja Data type instance.

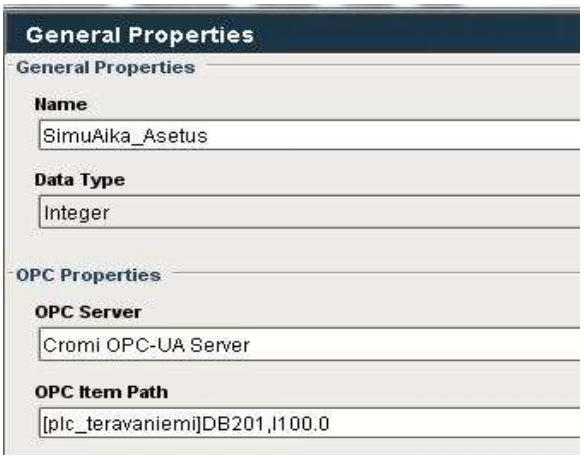
- OPC-tagi saa arvonsa OPC-palvelimen kautta eli yleensä PLC:ltä.
- Expression-tagi eli lauseketagi saa arvonsa laskennasta. Tätä tagia voidaan käyttää esimerkiksi laskemaan OPC-tagien arvoja uudestaan.
- Query-tagi saa arvonsa tietokannasta SQL-komennon avulla.
- Memory-tagi on muistitagi ja näihin valvomon suunnittelija voi esimerkiksi tallentaa jonkin vakioarvon, jota voi käyttää hyväksi valvomoa tehtäessä.
- Data type instance on tagiluokka. Esimerkiksi jos projekti sisältää aina samanlaiset tagit eri moottoreilla, voidaan tageista tehdä luokkakirjasto. Näin saadaan jokaiselle eri moottorille oma tagikirjastonsa. Tämä helpottaa myöhemmin tagien sijoittamisessa valvomoon, kun ne ovat hyvässä järjestyksessä. (Inductive University 2018s, viitattu 26.1.2018.)

Yleisin Ignitionissa käytetty tagi on OPC-tagi, joka saa arvonsa logiikalta käyttäen OPC-tiedonsiirron standardia. Niiden luominen yksi kerrallaan onnistuu Ignitionin tagien muokkaustyökalulla (kuva 4). Tärkeimmät ominaisuudet, jotka OPC-tagille määritetään ovat nimi, datatyyppi, OPC-palvelin ja OPC-polku. OPC-palvelin valitaan yleensä Gatewaylle konfiguroiduista palvelinyhteyksistä. OPC-polku kertoo tagin yhteysreitin ja se alkaa yleensä aina logiikan nimellä, joka Gatewaylle on luotu. Eri ohjelmoitavien logiikoiden välillä OPC-polun merkintätapa tagiin vaihtelee. (Inductive University 2018i, viitattu 26.1.2018.)



KUVA 4. Tagien muokkaustyökalu

OPC-polku jatkuu logiikan nimen jälkeen logiikan datalohkon numerolla, datatyypin tunnuksella ja logiikan osoitteella. Jokaiselle tagille valitaan oma datatyyppi, joka kertoo siihen tulevan arvon tyypin. Datatyyppi voi olla esimerkiksi kokonaisluku, reaaliluku tai binäärinen. Tagin OPC-polkuun syötettävän datatyypin tunnuksen pitää vastata tagille määritettyä datatyyppiä, jotta se saa arvonsa ongelmitta. Kuvassa 5 on esimerkki, jossa on luotu Integer-tyyppinen tagi. Integer-tyyppinen tagi saa vain kokonaisluku arvoja. Koska tagin datatyyppi on Integer, kuvasta 5 voi huomata, että OPC-polkuun logiikan osoitetta ennen tulee kirjain ”I” merkiksi siitä, että kyseessä on Interger-tyyppinen tagi. (Sama.)



KUVA 5. Integer-tyyppisen tagin ominaisuuksia

Tagit on tärkeä pitää järjestyksessä. Tämä onnistuu tekemällä pää- ja alikansiota, jonne tagit sijoitetaan ja nimetään kuvaavalla tavalla. Hyvin tehty kansiorakenne auttaa mahdollista seuraavaa valvomon ohjelmoijaa hahmottamaan tagit paremmin. Kansiorakenteen sekä tagien nimet on hyvä olla ensin kunnossa, ennen kuin tageja aletaan liittää ikkunoiden objekteihin. Tämä sen takia, että jos tagien polkuja tai nimiä muokataan jälkikäteen, objektit eivät enää löydä tagien arvoja, jotka niihin on jo liitetty. Tagien kansiorakenteen suunnitteluun kannattaa tämän takia käyttää aikaa, jotta vältytään ylimääräiseltä työltä tulevaisuudessa. (Inductive University 2018m, viitattu 26.1.2018.)

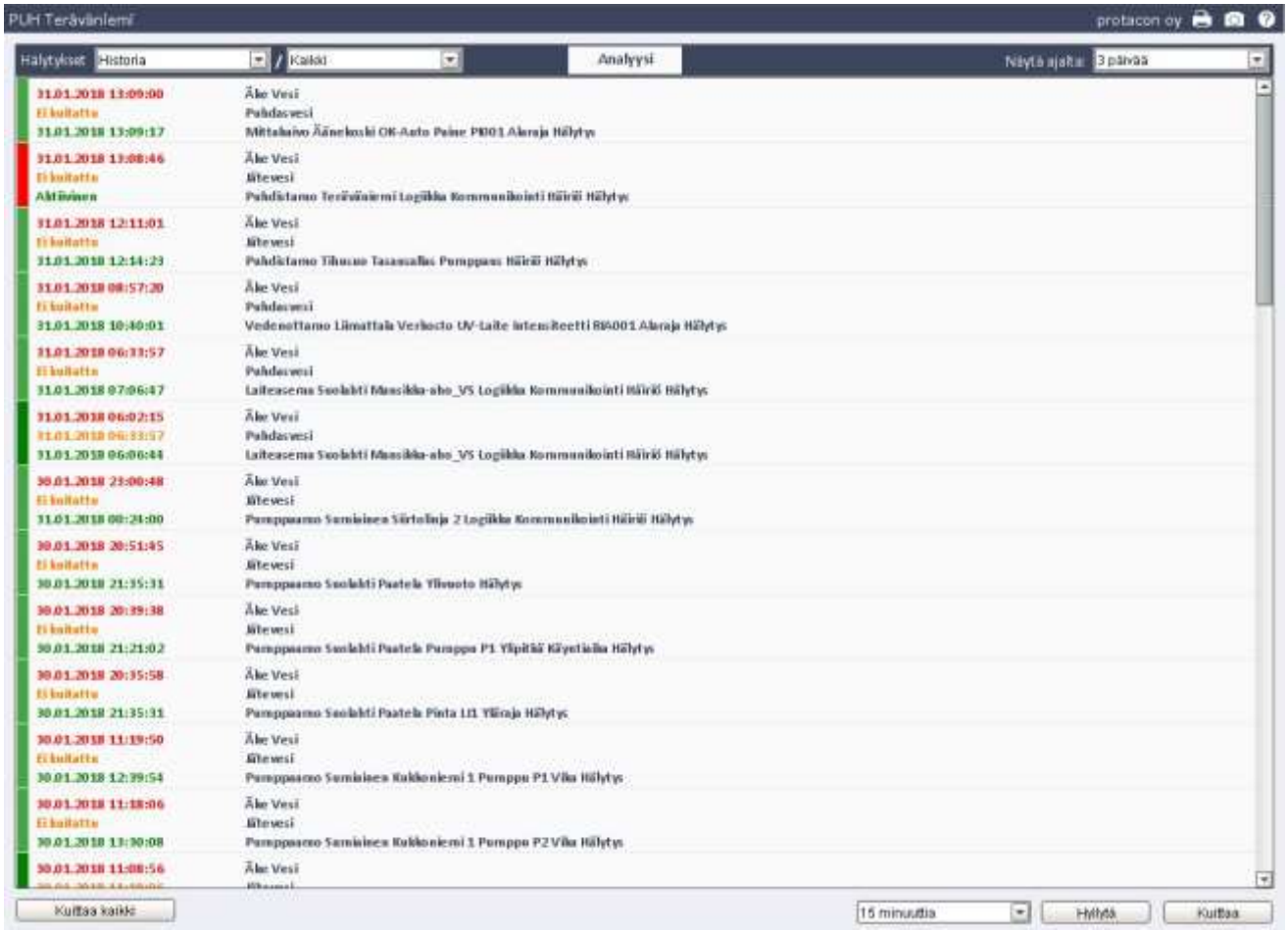
3.3.4 Hälytykset

Ignitionissa hälytykset liitetään tageihin tekemällä niihin hälytysominaisuus. Yhteen tagiin voi tehdä monia eri hälytyksiä. Jokainen hälytys tulee konfiguroida niin, että se toimii halutulla tavalla. Tärkeimpiä asetuksia, mitä hälytysten luonnissa on otettava huomioon, ovat sen nimi, kuvausteksti, asetusarvo ja prioriteetti, joka kertoo hälytyksen tärkeyden. Prioriteetin voi määrittää viidestä eri tärkeysluokasta. Hälytykselle on myös tärkeä asettaa raja-arvo, jonka saavutettaessa hälytys aktivoituu. Ignitionissa hälytyksen tilaa on mahdollista esittää sekä kuitata ja niitä on mahdollista lähettää eteenpäin esimerkiksi tekstiviestillä. (Inductive University 2018f, viitattu 26.1.2018.)

Käyttöliittymässä hälytyksiä on mahdollistaa esittää eri tavoin. Jokaisessa hälytyksen omaavassa tagissa on tila nimeltä “Alert Active”, joka siis kertoo, onko tagin jokin hälytys aktiivinen vai ei. Tätä ominaisuutta voi käyttää hyväksi indikoidessa hälytyksiä ikkunoiden objekteissa. Esimerkiksi punaisen ympyrän näkyvyyteen voi tehdä ehdon, että kun “Alert Active”-ominaisuus on tosi, ikkunaan ponnahtaa punainen ympyrä, joka kertoo operaattorille siitä, että hälytys on aktiivinen. (Inductive University 2018j, viitattu 26.1.2018.)

Hälytyksiä on myös mahdollista kerätä historia tietokantaan. Tämä tapahtuu luomalla Gatewaylle Alarm Journalin eli hälytysten lokikirjan, joka tallentaa toteutuneet hälytykset määritettyyn tietokantaan. (Inductive University 2018g, viitattu

26.1.2018.) Kun hälytyksiä tallennetaan tietokantaan, on niitä helppo selata käyttöliittymän puolella esimerkiksi taulukossa. Ignitionissa on valmiita hälytyksille tarkoitettuja taulukkokomponentteja, mutta myös normaaleja taulukoita löytyy, joihin hälytykset voi myös integroida. Kuvassa 6 oleva hälytystaulukko on käytössä jätevedenpuhdistamon valvomossa Äänekoskella. Se on esimerkki normaalista taulukosta, jossa näytetään sekä aktiiviset että tietokannassa olevat toteutuneet hälytykset. Taulukon kautta operaattori voi kuitata tai hyllyttää hälytyksiä. Hälytyksiä on myös mahdollista filtteröidä taulukon sisällä, mikä helpottaa operaattoria löytämään haluamansa hälytykset taulukosta.



KUVA 6. Äänekosken jätevedenpuhdistamon valvomon hälytystaulukko

3.3.5 Tietokanta ja historiankeruu

Projektien tietokannat konfiguroidaan Ignition Gatewaylle. Sieltä löytyy yleisimmin käytettyjen tietokantayhteyksien ajurit, kuten esimerkiksi MySQL:än ja Oraclen. Jokaiselle projektille valitaan aina yksi oletus tietokanta, johon kaikki historiatiedot tallennetaan. Historiatietoja käytetään yleisesti esimerkiksi kaavioissa tai taulukoissa kertomaan operaattorille jonkin prosessissa olevan laitteen tai mittauksen historiadataa.

Itse historiankeruu tapahtuu Ignitionissa hälytysten tavoin tagien ominaisuuksia muokkaamalla. Tageihin luodaan historiaominaisuus, jossa sille määritetään tietokanta, johon arvot halutaan tallentaa. Tagien arvot tallentuvat tietokannan taulukoihin, jotka Ignition luo automaattisesti, kun arvoja ensimmäisen kerran tallentuu tietokantaan. (Inductive University 2018p, viitattu 26.1.2018.)

Historia-arvojen tallennustahtia voi myös muuttaa. Gatewayllä tagille voi määrittää oman skannausluokan, joka kertoo tagille sen, miten nopeaan tahtiin arvoja tallennetaan tietokantaan. Historiaominaisuudessa tagille määritetään myös raja-arvo, mikä tarkoittaa sitä, että kun tagin arvo muuttuu raja-arvon verran, niin vasta silloin arvo tallentuu tietokantaan. (Sama.)

3.3.6 Omatekoiset objektit

Projektissa on yleistä, että samaa objektia kuten esimerkiksi moottoria tai säiliötä käytetään monesti läpi projektin. Näitä objekteja ei kuitenkaan tarvitse luoda jokaiselle ikkunalle omaansa. Ignitionissa on olemassa työkalu, jolla objekteista voi rakentaa omia pohjia. Objektin voi tämän jälkeen tallentaa ja sitä voi käyttää uudestaan. Omatekoisia objekteja voi luoda joko projektin omaan kansioon, mikä tarkoittaa sitä, että objekteja voi käyttää vain projektin sisällä. Toinen vaihtoehto on luoda objektit globaaliin kansioon, jolloin tehtyjä objekteja voi käyttää myös muissa valvomoprojekteissa, jotka ovat saman Ignition-palvelimen alla. (Inductive University 2018b, viitattu 26.1.2018.)

Objekteihin voi luoda mukautettuja ominaisuuksia, joita voi käyttää hyväksi objektin datalähteen määrittämiseksi tai objektin muokkaamisen apuna. Datalähteen on tärkeä olla kunnossa, jotta objektit saavat arvonsa oikeista kansioista ja tageista. (Sama.)

Omatekoiset objektit luodaan pääpohjalle. Tämä tarkoittaa sitä, että kun pohjaan tulee muutoksia, niin kaikki ikkunoille jo raahatut objektit päivittyvät. Näin ollen valvomosovelluksen suunnittelijan ei tarvitse muokata jokaista objektia yksitellen, vaan objektien muokkaaminen tapahtuu helposti ja ennen kaikkea nopeasti objektin pääpohjaa muokkaamalla. (Sama.)

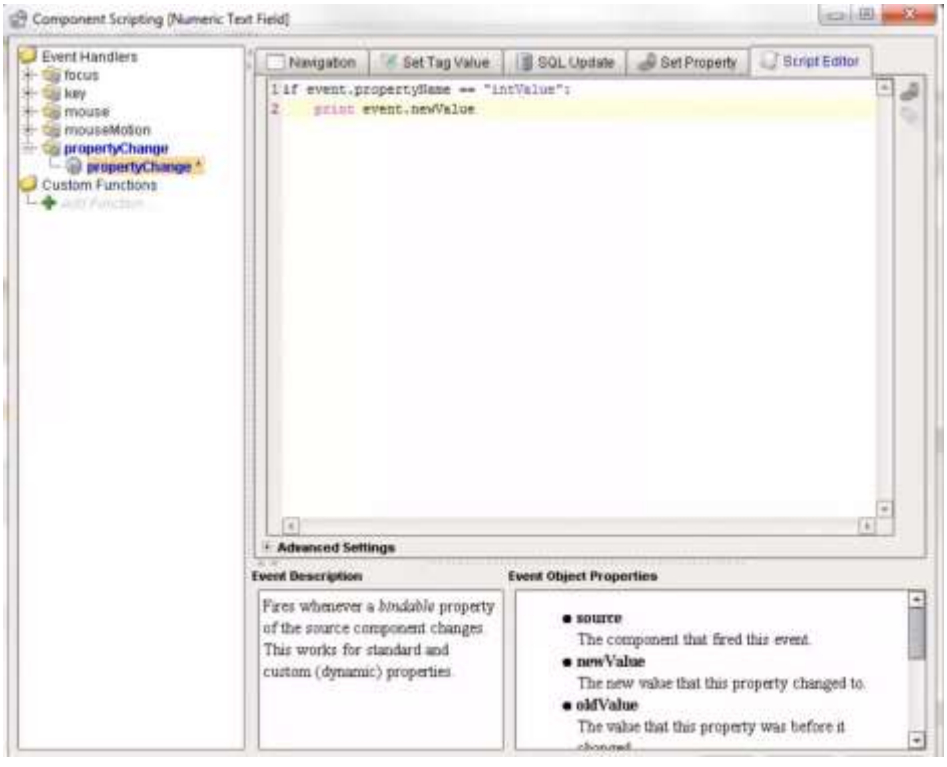
3.4 Ohjelmointi ja objektien sisäiset toiminnot

Ohjelmointia Ignitionissa voi käyttää monessa eri paikassa, ja sillä on mahdollista tehdä erilaisia toimintoja esimerkiksi objekteihin, ikkunoihin tai tageihin. Ignitionissa koodauskielenä toimii Python ja siitä versio 2.5. Jos Ignitionin omilla sisäänrakennetuilla toiminnoilla ei saada haluttua lopputulosta, yleensä on turvauduttava koodaamiseen. (Inductive University 2018o, viitattu 29.1.2018.) “Python on korkeatasoinen, tulkattu, interaktiivinen ja objektipainotteinen ohjelmointikieli” (Python overview 2018, viitattu 29.1.2018). Se sopii hyvin aloitteleville koodareille ja se on monikäyttöinen sekä helppolukuinen. Sitä käytetään muun muassa GUI-applikaatioissa (Graphical user interface) eli graafisissa käyttöliittymissä ja pelien tekemisessä. (Sama.)

Ignitionissa ohjelmointi tapahtuu kuvan 7 mukaisessa ikkunassa. Kirjoitetulle koodille valitaan yksi tapahtuma, jolloin koodi suoritetaan. Tämä tapahtuma valitaan edellä mainitun kuvan vasemmassa reunassa olevasta “Event Handlers”-valikosta. Tapahtuma voi olla esimerkiksi hiiren klikkaus tai objektin ominaisuuden muutos. Objektit, joissa on käytetty koodaamista, näkyvät sovelluksessa koodaus ikonilla varustettuna. Tämä auttaa mahdollista seuraavaa valvomosovelluksen suunnittelijaa löytämään sovelluksessa käytetyt koodit ja näin ollen suunnittelijan on helpompi päästä perille siitä, miten käyttöliittymän toiminnot on rakennettu. (Inductive University 2018d, viitattu 29.1.2018.)

Kun suunnittelija on valinnut tapahtuman, jolloin koodi aktivoituu, tulee sen mukana myös sisäisiä muuttujia, joita voi käyttää hyväksi koodia kirjoittaessa. Jokaisella erilaisella koodin laukaisevalla tapahtumalla on omat sisäiset muuttujansa. Sisäiset muuttujat näkyvät kuvan 7 oikeassa alalaidassa olevassa valikossa. Näiden avulla koodin kirjoittaja voi tehdä esimerkiksi ehtoja koodiin käyttäen apuna if-lausetta. (Inductive University 2018l, viitattu 29.1.2018.)

Alla olevassa kuvassa 7 on esimerkki koodista, jossa sisäisiä muuttujia on käytetty hyväksi ehtolausekkeessa. Kuvan koodissa sen laukaisevana tapahtumana toimii objektin ominaisuuden muuttuminen, joka tarkoittaa sitä, että aina kun objektin ominaisuus muuttuu, niin koodi suoritetaan. Kuvan 7 koodissa määritellään, että jos muuttunut ominaisuus on numeerisen tekstikentän arvo, niin koodi tulostaa muuttuneen uuden arvon. Koodissa on siis käytetty hyväksi kahta erilaista sisäistä muuttujaa, jotka ovat nimeltään “propertyName”, joka siis kertoo, mikä ominaisuus objektissa muuttui ja “newValue”, joka kertoo muuttuneen uuden arvo. (Inductive University 2018l, viitattu 29.1.2018.)



KUVA 7. Ignitionin koodausikkuna ja esimerkkikoodi, jossa on käytetty hyväksi sisäisiä muuttujia (Inductive University. 2018l)

Objektien toimintoja ei aina tarvitse koodata, vaan Ignitionissa on myös saatavilla muutama sisäänrakennettu ominaisuus. Koodaamisen lisäksi objekteille on mahdollista tehdä seuraavia toimintoja: navigointi, tagiin kirjoittaminen, objektin ominaisuuden muuttaminen tai SQL-komento. Nämä toiminnot näkyvät kuvan 7 ylälaidassa. Yhdelle objektille voi valita vain yhden toiminnon ja sille voi valita yhden laukaisevan toiminnon kuten python koodissakin. (Inductive University 2018n, viitattu 29.1.2018.) Nämä sisäänrakennetut toiminnot riittävät yleensä ainoastaan yksinkertaisiin ja yksittäisiin toimintoihin. Näin ollen, jos halutaan tehdä monimutkaisempia sekä monia toimintoja samanaikaisesti, niin on käytettävä koodaamista.

4 VALVOMON VISUAALISET OMINAISUUDET

On tärkeää, että operaattori pystyy tulkitsemaan valvomon ikkunoiden välittämää sanomaa. Tämän saavuttamiseksi näytöistä pyritään tekemään mahdollisimman selkeitä ja hahmotettavia. Näyttöjen tulkinta on jokaisella meillä erilainen. Tulkintaan vaikuttavat muun muassa ihmisen fysiologiset ominaisuudet ja näkökyky. (Metsämäki 1995, 7.)

Fysiologisten ominaisuuksien lisäksi merkittävä tekijä sille, miten tulkitsemme käyttöliittymiä, on ihmisten oma ympäristömalli. Malli muokkautuu sen mukaan, mitä olemme omassa kulttuurissamme tottuneet näkemään, ja siitä, miten nähtyjä asioita on tulkittu. Käyttöliittymän suunnittelijan vaikeimpia tehtäviä onkin tehdä käyttöliittymästä sellainen, että se olisi juuri operaattorille selkeä ja ymmärrettävä. Parhaiten tässä tehtävässä kuitenkin onnistuu, kun käyttöliittymän suunnittelussa käytetään vain yleisesti tunnettuja elementtejä ja symboliikkaa, jotta vältytään vääriltä tulkinnoilta. (Sama, 6.) “Mitä enemmän meillä on tekijän kanssa yhteinen symboliikka, sanavarasto, sitä paremmin sanoma rakentuu mielessämme alkuperäiseen lähtömuotoonsa” (Sama, 7).

Tässä luvussa käydään läpi visuaalisesti hyvin tehdyn käyttöliittymän ominaispiirteitä. Luvussa kerrotaan muun muassa käyttöliittymän symboleista, objektien sijoittelusta, väreistä ja tekstistä.

4.1 Symbolit

Symbolit ovat käyttöliittymän kuvakkeita ja ne voivat olla esimerkiksi yksittäisen laitteen yleisesti tunnettu symbolikuva. Niiden tarkoitus on informoida operaattoria prosessissa vallitsevista olosuhteista. Symbolilla on aina vain yksi merkitys, jonka takia käyttöliittymässä on hyvä käyttää vain yleisesti tunnistettavia symboleita, jotta operaattorin on helpompi tulkita niiden merkitystä. Jos vieraampaa symbolia käytetään, se on hyvä selventää vielä erikseen tekstillä, mutta yleisesti symbolin tulisi olla niin kuvaava, että se ei tarvitse erillistä tekstiä kertomaan, mikä

se on. Symbolit toimivat käyttöliittymässä paremmin kuin pelkkä teksti, koska niitä mielekkäämpi ja nopeampi käyttää. (Vehmas 2012, 15–16.)

Käyttöliittymässä on hyvä pyrkiä samaa asiaa kuvaavan symbolin samankaltaisuuteen läpi koko valvomon. Yleensä valvomoihin tehdään jokaiselle symbolille oma pohja, jota käytetään aina tarpeen tullen. Symbolien samankaltaisuus lisää käyttöliittymän tulkittavuutta. Symbolien merkitystä on helppo tuoda enemmän esiin lisäämällä niihin kuvaavia pikaohjeita, jotka tulevat esimerkiksi näkyviin, kun operaattori vie hiiren kursorin symbolin päälle. Symboleita suunniteltaessa on otettava huomioon niiden kulttuurilliset erot, koska ne saattavat olla loukkaavia tai väärin ymmärrettäviä eri maissa ja kulttuureissa. (Sama, 15–16.)

Käyttöliittymään symbolit ovat usein SVG-grafiikkaa eli vektorigrafiikkaa. SVG-grafiikka on XML:ään perustuva tekstipohjainen grafiikkakieli. Tässä kielessä symbolin jokainen elementti on omansa ja niitä voi muokata haluamallaan tavalla. Muokattavia ominaisuuksia ovat esimerkiksi muoto, väri ja sijainti. “Toisin kuin pikseleistä koostuvat bittikarttakuvat, vektorigrafiikka perustuu koordinaatistoon sidottuihin geometrisiin primitiiveihin, kuten pisteet, viivat, ympyrät, (bezier-) käyrät ja monikulmiot, jotka kaikki perustuvat matemaattisiin yhtälöihin” (Kotisalo 2007, 3). Matemaattisuuden ansiosta vektorigrafiikoiden kokoja ja muotoja voi muokata ilman, että kuvanlaatu kärsii. Vektorigrafiikka on resoluutioriippumaton, mikä mahdollistaa sen, että kun kuvia suurennetaan, ne eivät näytä karkeilta. (Sama, 3.)

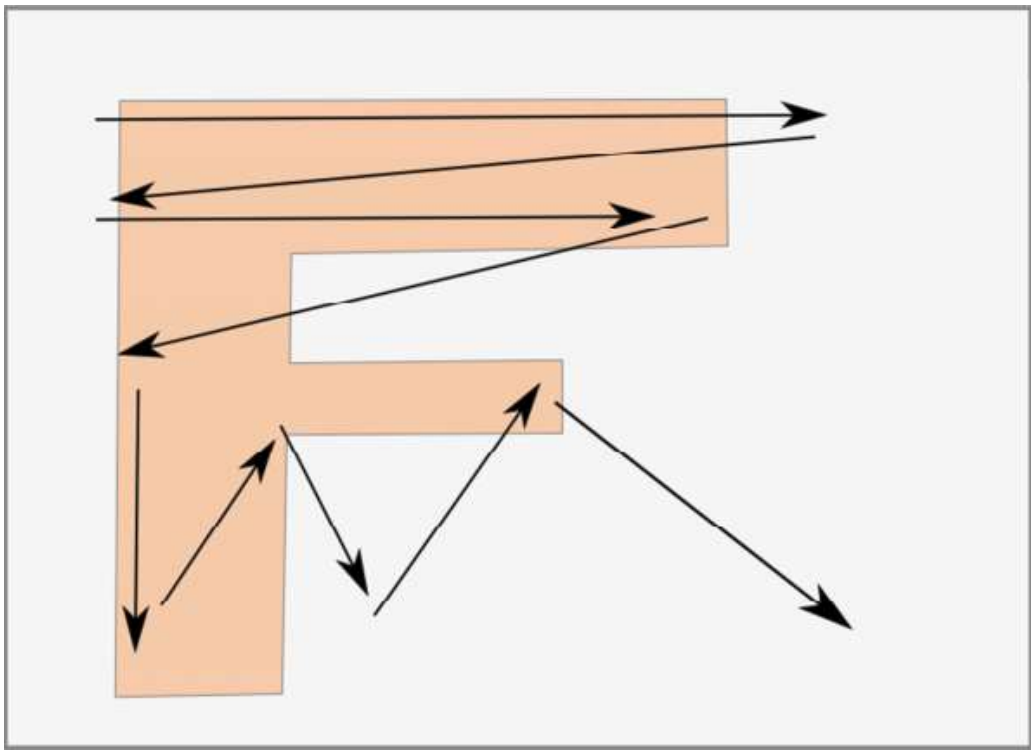
SVG-grafiikan ansiosta symboleita on mahdollista animoida. Tämä on kuitenkin yleensä vältelty ominaisuus ainakin teollisuuden valvomoissa. Animoidut objektit tekevät käyttöliittymästä helposti sekavamman ja ne saattavat rasittaa operaattoria, joka johtaa käyttöliittymän vaikeaan tulkitsemiseen. Animoinnin tarpeellisuutta kannattaa todella harkita ennen sen käyttämistä. (Vehmas 2012, 5.)

4.2 Sijoittelu

Oikeanlainen sommittelu ja ryhmittely tekee käyttöliittymästä selkeämmän ja tasapainoisemman, mikä tekee siitä mielekkäämmän käyttää. Ympäristö vaikuttaa ihmisiin ja se tuo meille erilaisia mielialoja. Esimerkiksi jos työhuone on sotkuinen, niin ei se yleensä ainakaan auta saavuttamaan hyvää mielialaa. Tämä sama pätee myös käyttöliittymissä. Mitä selkeämpi ja siistimpi käyttöliittymä on, niin sitä mukavampi sitä on käyttää. (Metsämäki 1995, 15.)

Valvomon ikkunoilla tulee olla aina riittävästi tyhjää tilaa. Se on yhtä tärkeää kuin täytettykin tila, koska se auttaa lisäämään käyttöliittymän selkeyttä. Sekava ja liian täyteen pakattu ikkuna välittää liikaa informaatiota, josta operaattori ei välttämättä pysty sisäistämään kaikkea. Tästä syystä tärkeitä prosessin muutoksia voi mennä operaattorilta aivan ohi, mikä voi johtaa kriittisissä kohteissa vaarallisiinkin tilanteisiin. (Metsämäki 1995, 40.)

Käyttöliittymästä tulisi aina pyrkiä rakentamaan mahdollisimman looginen ja käyttäjää ohjaava. On hyvä miettiä sitä, minne operaattori suuntaa katseensa ensin ja minne se sen jälkeen jatkuu, kun hän aukaisee valvomoikkunan. Länsimaisessa kulttuurissa lukusuunta on yleisesti vasemmalta oikealle ja ylhäältä alas. Tämän takia käyttöliittymän ikkunat on hyvä rakentaa niin, että tärkeimmät asiat ovat alla olevan kuvan (kuva 8) F-kirjaimen näköisen kuvion sisällä. Käyttäjä löytää lukusuunnassa ensimmäisenä vastaan tulevat objektit nopeiten, joten tärkeimpien asioiden sijoittamista ikkunaan kannattaa pohtia tämä asia huomioon ottaen. (Vehmas 2012, 3.)

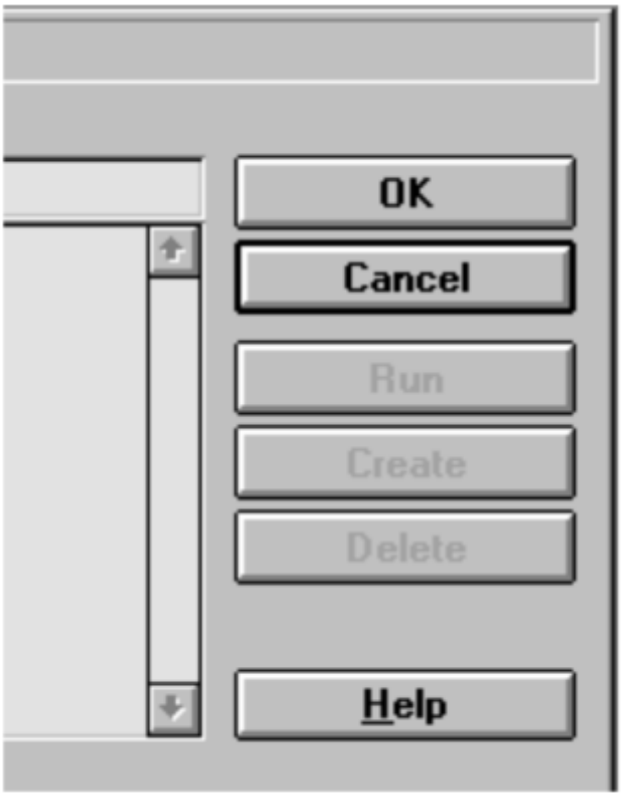


KUVA 8. Katseen eteneminen ja tärkeimmät kohdat ikkunalla länsimaisessa käyttöliittymässä (Vehmas 2012, 3)

Valvomon ikkunoiden loogisuutta lisää myös hyvä objektien ryhmittely. Tällä tarkoitetaan sitä, että samanmuotoiset sekä samaan asiaan liittyvät objektit sijoitetaan ikkunassa systemaattisesti samaan paikkaan lähelle toisiaan. Ryhmittely lisää käyttöliittymän käytettävyyttä. Operaattorin ei tarvitse etsiä objekteja niin kauaa, koska ne on ryhmitelty yksiksi kokonaisuuksiksi. Kokonaisuudet toimivat oikein, kun sen objektit liittyvät samaan asiayhteen, jolloin operaattori tietää löytävänsä tietyt asiat omista ryhmistään. Ryhmittely antaa samalla myös mahdollisuuden lisätä enemmän tietoa valvomoon ilman, että siitä tulisi sekava tai epämiellyttävä käyttää. (Metsämäki 1995, 20–25.)

Hyvä ryhmien koko on yleisesti 5–6 asiakokonaisuutta. Ryhmät kannattaa erottaa tarpeeksi riittävällä välillä toisistaan, jotta ne eivät sekoitu keskenään. Ryhmien välille tulevaa tyhjää tilaa kannattaa pitää saman kokoisena läpi valvomon, jotta käyttöliittymä säilyttäisi selkeytensä. Jos tyhjää tilaa ei ole mahdollista saada tarpeeksi ryhmien välille, niin apuna voi käyttää viivoja, joilla jokainen ryhmä voidaan laittaa omaan laatikkoonsa. (Sama, 20–25.) Alla on esimerkki (kuva 9) hyvästä ryhmittelystä. Kuvassa samaan asiayhteyteen liittyvät napit on sijoitettu lähelle

toisiaan sekä ryhmät on erotettu toisistaan riittävällä välillä. Kuvan ryhmittelystä tekee myös hyvää se, että napit ovat samanmuotoisia ja -kokoisia sekä ne on sommiteltu hyvin samaan linjaan pystysuunnassa.



KUVA 9. Esimerkki hyvästä objektien ryhmittelystä (Metsämäki 1995, 23)

Jos valvomo vaatii paljon tiedon välittämistä eikä ryhmittely pelkästään saa käyttöliittymästä tarpeeksi jäsenneltyä, niin on syytä käyttää eri tasoja. Näissä tasoissa operaattori siirtyy käyttöliittymässä päätasolta seuraavalle tasolle ja siitä seuraavalle. (Sama, 20–25.) Tasoja voivat olla esimerkiksi avautuvat ponnahdusikkunat tai erilliset ikkunat, jotka aukeavat päätasolta. “Hakiessamme kirjaa kirjastosta nimen perusteella on järkevää mennä selaamaan ylimmän tason kortistosta nimen mukaan oikea kirjakortti. Sitten seuraavalla tasolla haemme kirjan hyllyn numeron perusteella ja lopuksi alimmalla tasolla etsimme kirjan hyllystä nimen perusteella” (Sama, 25).

4.3 Värit

Värit ja niiden vaihtuminen on tehokas keino saada operaattorin huomio. Esimerkiksi länsimaissa käytetään yleisesti punaista väriä kertomaan varoituksesta operaattorille. Keltaista käytetään herättämään huomio ja vihreää hyväksymisen merkkinä. Värien näkyvyys riippuu aina taustaväristä. Tummempi objekti näyttää tummemmalta vaalealla pohjalla kuin tummalla pohjalla. Tästä syystä käyttöliittymissä kannattaa suosia harmaan sävyisiä taustoja, koska niissä huomion herättämiseksi valitut kirkkaat värit näkyvät paremmin. Yleisimpiä taustavärejä ovat sininen, siniharmaa ja harmaa. Ihmisen silmä työskentelee nähdäkseen värejä, jonka takia kirkkaita värejä kannattaa välttää taustaväreissä. Tällä vältetään operaattorien silmien väsyminen valvomon ikkunoihin. Käyttöliittymään on hyvä valita yksi väriskaala, jota käytetään koko valvomossa. (Metsämäki 1995, 32–35.)

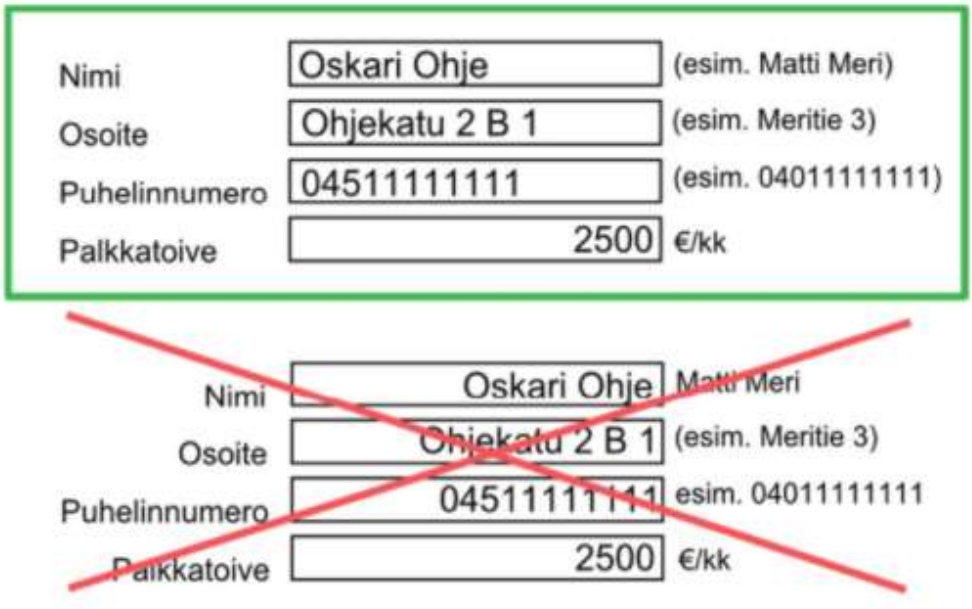
Värejä ei kuitenkaan koskaan saa käyttää ainoana viestin välittäjänä mahdollisen operaattorin värisokeuden takia. Värin viestiä tarkennetaan yleisesti vielä tekstillä. Perusväreistä punainen, keltainen ja sininen erottuvat toisistaan helposti, kun taas sininen ja vihreä sekä sininen ja punainen eivät. Tästä syystä näitä väripareja kannattaa välttää käyttämästä lähekkäin käyttöliittymässä. Nämä värit sekoittuvat helposti keskenään ja operaattori joutuu työskentelemään silmillään paljon erottaakseen värit toisistaan. Värin vaihdon lisäksi huomiota voi herättää objektin kokoa, muotoa, suuntaa tai varjostusta muuttamalla värin vaihdon yhteydessä. Tehokas keino huomion herättämiseksi on myös värien vilkuttaminen, jota kannattaa kuitenkin käyttää harkiten, koska se voi olla liiankin silmään pistävä ja häiritsevä. (Sama, 26.)

Perusväreillä on aina eri merkitykset eri kulttuureissa. Tämän takia on hyvä ottaa selvää valvomon kohdemaan väritottumuksista, ennen kuin alkaa tehdä käyttöliittymää. Euroopassa käytetään yleisesti hillittyjä värejä, kun taas Amerikassa käytetään monipuolisesti erilaisia värejä. (Sama, 36–37.)

4.4 Teksti

Käyttöliittymässä on hyvä käyttää enintään kahta eri fonttityyppiä. Otsikoiden ja normaalin tekstin välillä käytetään erilaisia kokoja. Tekstin on tärkeää olla tarpeeksi suurta, jotta se näkyy käyttäjälle mahdollisimman hyvin. Luettavassa tekstissä kannattaa käyttää Arial-tyyppistä fonttia ja pitkissä teksteissä päätteellistä Times New Roman -tyylistä fonttia. (Metsämäki 1995, 8.) Yleisimpiä paikkoja, missä tekstiä näkyy valvomossa ovat teksti- ja numeeriset tekstikentät, otsikot, yksiköt ja taulukot.

Tekstikentän otsikon voi sijoittaa joko kentän päälle tai vasemmalle puolelle. Jos otsikot ovat kentän vierellä, niin tasataan ne vasemmalle. Alla olevassa kuvassa 10 on esimerkki siitä, miten tekstit on hyvä sijoittaa tekstikentissä. Kuvasta voi huomata, että tekstit tasataan kentän vasempaan laitaan ja numerot oikeaan. Poikkeuksena puhelinnumerot, jotka tasataan vasemmalle. Numerot ovat oikealla sen takia, että ne ovat mahdollisimman lähellä yksikköä, joka laitetaan yleisesti kentän oikealle puolelle. Kentille on hyvä merkitä yksiköt, jotta operaattori tietää, mitä arvoa mikäkin kenttä koskee. Yksiköiden ja muidenkin tekstien sijoittelussa kannattaa pyrkiä käyttämään yhtä tyyliä läpi valvomon, jotta valvomon selkeys säilyy. (Sama, 9.)



KUVA 10. ”Hyvä ja huono esimerkki tekstikentistä” (Vehmas 2012, 8)

Valvomossa pitää pyrkiä yhtenäiseen kieleen ja lainasanoja on vältettävä. Yleisimpiä lainasanoja ovat on, off, start ja stop. Nämä tulisi korvata suomenkielisillä nimityksillä. Tekstin suunta kannattaa aina pyrkiä pitämään horisontaalisena, koska vertikaalista tekstiä on vaikea lukea. Lyhenteiden käyttöä tulisi yleisesti välttää, koska ne voivat olla käyttäjälle tuntemattomia, mistä voi syntyä väärinymmärryksiä. Poikkeuksena yksiköt ja viikonpäivät, jotka ovat yleisesti tunnettuja. (Sama, 9-11.)

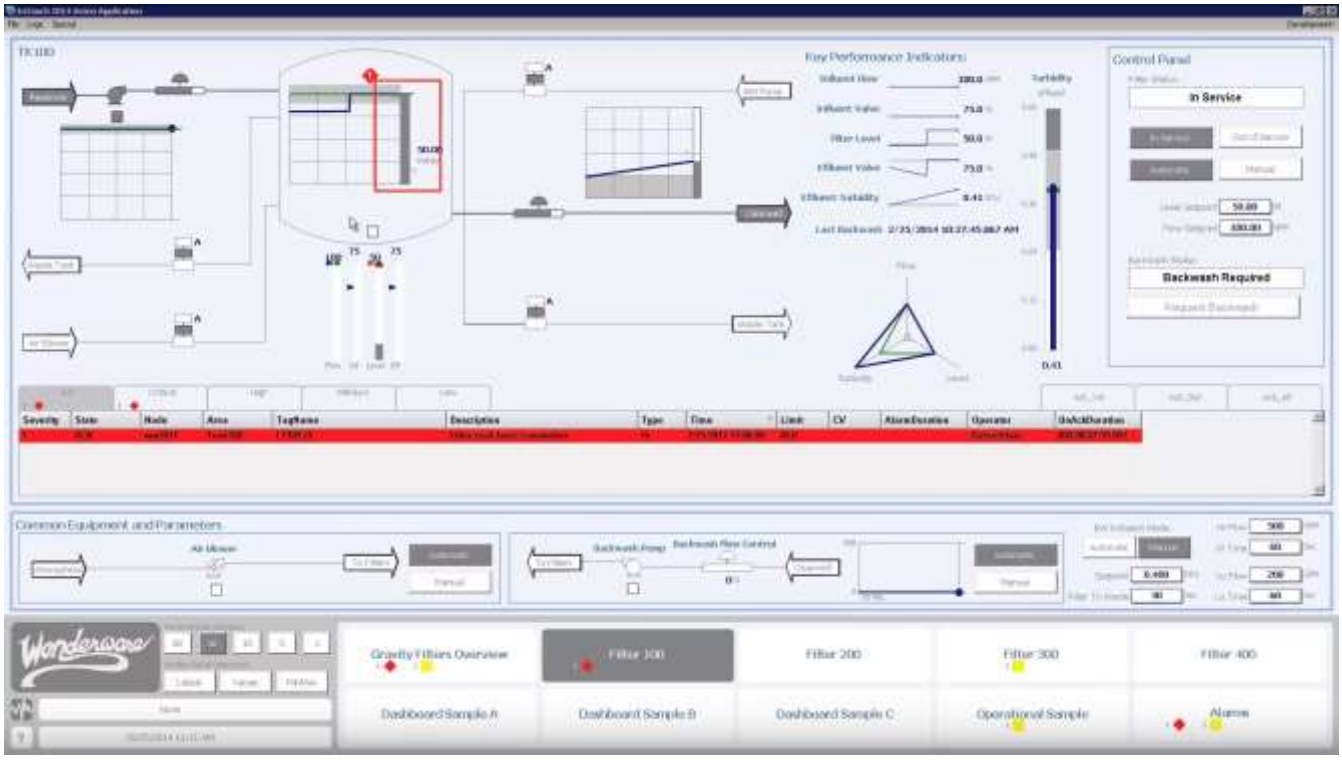
4.5 Tilannetietoisuus

”Situational awareness” eli tilannetietoisuus on käyttöliittymän suunnittelussa käytetty trendi. Sen pääperiaatteena on tehdä visuaalisesti selkeä käyttöliittymä. Sillä pyritään helppoon käyttöliittymän kouluttamiseen ja ongelmien nopeaan reagoimiseen, jotka molemmat tekevät tuotantolaitoksesta turvallisemman ja näin ollen tuottavamman. (Schneider Electric Software 2014, viitattu 5.2.2018.)

Nykyään valvomoihin tulevan tiedon määrä vain kasvaa ja kasvaa. Tämän takia operaattorilta vaaditaan suuren tietomäärän sisäistämistä. ”Situational awareness” pyrkii ottamaan tämän huomioon ja helpottamaan käyttöliittymän käyttöä operaattorille. Trendissä pyritään vähentämään tietomäärän sekavuutta keräämällä vain tärkeät asiat ja jaottelemalla niitä sinne missä niitä tarvitaan, ja milloin niitä tarvitaan. Käyttöliittymät pyritään rakentamaan käyttäjän tarpeiden mukaiseksi. Operaattoria pyritään ohjaamaan mahdollisimman paljon, joka vähentää käyttöliittymän stressaava käyttöä. Mahdollisista epänormaaleista tiloista prosessissa ilmoitetaan selkeästi niin, että ne eivät häviä ruudulla muiden näkyvien asioiden joukkoon. (Sama.)

Alla on esimerkki (kuva 11) siitä, miltä näyttää käyttöliittymä, jossa on käytetty hyväksi ”Situational awarenessin” periaatteita. Tilannetietoisuutta käyttöliittymässä lisää taustaväri, joka on tärkeä valita niin, että hälytykset näkyisivät näytöllä paremmin. Kuvassa taustaväri on siniharmaa, jonka ansiosta punaisella näkyvät hälytykset huomataan helpommin. Hälytysten näkyvyyttä lisää myös se, että ne laatikoidaan punaisella ja se, että käyttöliittymässä on ylipäätään vältelty

käyttämästä liikaa erilaisia värejä. On tärkeää, että punaista väriä ei käytetä missään muualla kuin hälytyksissä. Näin ollen operaattori ei mene sekaisin siitä, onko tieto hälytys vai ei. (Wonderware HMI SCADA 2014, viitattu 5.2.2018.)



KUVA 11. Hälytysten näyttäminen tilannetietoisessa käyttöliittymässä (Wonderware HMI SCADA 2014)

5 VALVOMOSOVELLUKSEN TOTEUTTAMINEN

Jätevedenpuhdistamon valvomosovellus tehtiin käyttäen Ignition SCADA -ohjelmaa. Uuden valvomosovellus osion tuli sisältää muun muassa uudet tagit, ikkunat, kaaviot, hälytysnäytön, raportit, käyttöpäiväkirjan ja historiatiedon keruun. Näistä muun muassa raportit, käyttöpäiväkirja, hälytysnäyttö ja osa kaavioista oli luotu jo valmiiksi, jonka takia niitä ei tehty uudestaan tai muokattu.

Työ oli hyvä aloittaa tutustumalla käytettävään Ignition-ohjelmaan, jotta sen periaatteet sekä perustoiminnot tulisivat tutuiksi. Ignitionin omilta nettisivuilta löytyi kattavia opastusvideoita, joista oli paljon apua käyttöliittymää tehtäessä. Ohjelmasta on mielestäni tärkeää saada hyvä yleiskuva, jotta työn aloitus olisi helpompaa. Valvomosovelluksesta tuli tehdä mahdollisimman samannäköinen kuin jo olemassa oleva vanha käyttöliittymä. Vanha sovellus toimi pohjana työlleni ja käytin sitä myös paljon hyväksi omassa työssäni. Osa sovelluksen toiminnoista on kopioitu vanhasta ja muokattu niin, että ne toimisivat myös tekemäni sovelluksen ja tagien kanssa.

Tässä luvussa kerron työni eri vaiheet sekä pääperiaatteet siitä, kuinka valvomosovellus toimii. Käyn läpi muun muassa valvomon ikkunat, tagien luonnin sekä niiden sijoittelun ja valvomon toiminnallisuudet.

5.1 Valvomoikkunat ja niiden piirto

Valvomossa olevat ikkunat piirrettiin tilaajalta saatujen PI-kaavioiden pohjalta. Ikkunat pyrittiin tekemään niin, että ne muistuttavat jo vanhassa ohjelmassa olevia ikkunoita. Vanhasta ohjelmasta löytyviä objekteja, kuten altaita ja putkia käytettiin mallina ikkunoita piirrettäessä. Myös prosessilaitteet kuten venttiilit, moottori ja kompressorit on tehty vanhoja pohjia muokaten. Kun vanhaan ohjelman lisätään uutta osiota, on hyvä, että se muistuttaa vanhaa, jotta valvomon käyttäjän on helpompi sopeutua uuteen käyttöliittymään. Ikkunoita on päivitetty ja korjailtu työn edetessä.

5.1.1 Pääikkunat

Pääikkunoiden tarkoitus on antaa yleiskuva prosessista sekä informoida käyttäjää prosessissa vallitsevista olosuhteista ja tilojen muutoksista. Valvomosovellus koostuu kolmesta eri pääikkunasta, jotka ovat päänäyttö, välipumppaamo ja ilmastusaltaat sekä jälkiselkeytys. Ikkunat on piirretty ja väritetty tilaajalta saatujen PI-kaavioiden mukaisiksi.

Päänäytössä (liite 7) on esitetty koko jätevedenpuhdistusprosessi sekä kaikista merkityksellisimmät mittaukset osaprosessien varrelta, jotka tilaaja on katsonut tärkeimmäksi näyttää. Näin ollen operaattori huomaa heti päänäytöstä, jos jokin kriittinen osio jossain osaprosessissa ei pysy halutuissa rajoissa. Muut pääikkunat eli “Välipumppaamo & Ilmastusaltaat”- ja “Jälkiselkeytys”-ikkunat sisältävät nimensä mukaisen osaprosessin kaikki automaatiojärjestelmään kytketyt mittaukset ja prosessilaitteet. Liitteissä 2–4 on valvomosovellukseen tulevien osaprosessien PI-kaaviot ja liitteissä 5–7 on kuvat tekemistäni valvomon pääikkunoista. PI-kaavion kuvat välipumppaamosta ja ilmastusaltaista on rakennettu yhteen valvomoikkunaan.

Pääikkunoiden rakentaminen aloitettiin laittamalla kuviin tarvittavat objektit. Kuviin haettiin objekteja vanhasta käyttöliittymästä, jotka sijoitettiin sekä mitoitettiin vastaamaan prosessia. Myös pohjia tarvittavista prosessilaitteista ja mittaus indikaattoreista haettiin vanhan ohjelman objektipohjien-kirjastosta.

Kun kaikki tarvittavat objektit saatiin tuotua kuvaan, niin alkoi objektien sijoittelu sekä sommittelu. Objektien sijoittelussa pyrittiin siihen, että tyhjää tilaa objektien välillä jäisi riittävästi, koska tämä tekee käyttöliittymästä sujuvamman käyttää sekä selkeämmän. Liian tiheään sekä huonosti sommitellut objektit ovat käyttäjälle vaikealukuisia, joka voi johtaa siihen, että jotain tärkeää informaatiota, mitä käyttöliittymä pyrkii välittämään saattaa mennä kokonaan käyttäjältä ohi.

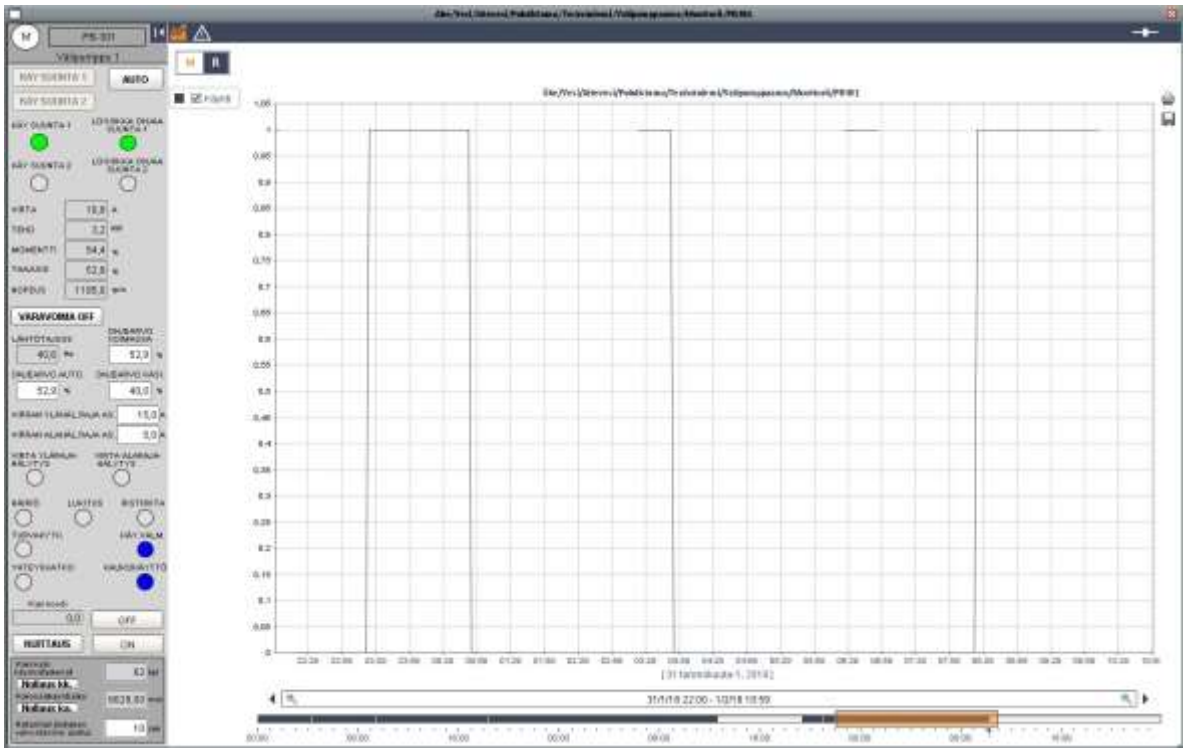
Objektien sijoittelun jälkeen alkoi putkilinjojen sekä muiden apuviivojen piirto. Näiden piirrossa on käytetty Ignitionin omaa viivatyökalua. Viivojen piirtojen jälkeen putkia kuvaavien viivojen paksuutta muutettiin paksummiksi, jotta operaattori

erottaa putkilinjat muista viivoista. Tieto siitä, mistä jätevesi on tulossa ja minne se on menossa, on toteutettu niin, että linjojen päälle tai alle on kirjoitettu teksti veden tulo- tai menokohteesta. Putkien päihin on myös piirretty suuntaa antava kolmio, jotta käyttäjän on helppo seurata, mihin suuntaan jätevesi kulkee prosessissa.

Lopuksi prosessilaitteisiin sekä mittauksiin on laitettu niiden positiotunnukset käyttäen apuna tunnustyökalua. Tällä työkalulla voi raahata ikkunaan tekstikentän, johon voi vapaasti kirjoittaa tekstiä. Näin käyttäjä tietää, mikä ikkunalla oleva objekti vastaa mitäkin laitetta tai mittausta. Osaan objekteista positiotunnukset on tehty kirjoittamalla se objektin mukautettuun ominaisuuteen nimeltä ”HeaderText” eli otsikkoteksti (ks. luku 2.2.6), joka on liitetty objektipohjan otsikoksi tarkoitettuun tekstikenttään.

5.1.2 Ponnahdusikkunat

Käyttöliittymässä prosessialtailla, prosessilaitteilla ja mittauksilla on omat ponnahdusikkunansa ja ne aukeavat joko pääikkunoilla olevista napeista, prosessilaitteista tai mittausindikaattoreista klikkaamalla. Ponnahdusikkunoiden tarkoitus on antaa enemmän tietoa valitusta kohteesta sekä antaa operaattorille mahdollisuus operoida niitä. Kuten kuvasta 12 voi huomata, niin pumpun ajaminen sekä sen asetusarvojen muutokset tapahtuvat sen omalla ponnahdusikkunalla. Ikkunat sisältävät erilaisia nappeja, numeerisia tekstikenttiä, tekstikenttiä, indikointivaloja ja käyrästön, jossa näkyy kohteen historia- sekä reaaliaikaisia arvoja. Ponnahdusikkunoiden objektit on tehty käyttäen Ignitionin omia työkaluja (ks. kuva 3).

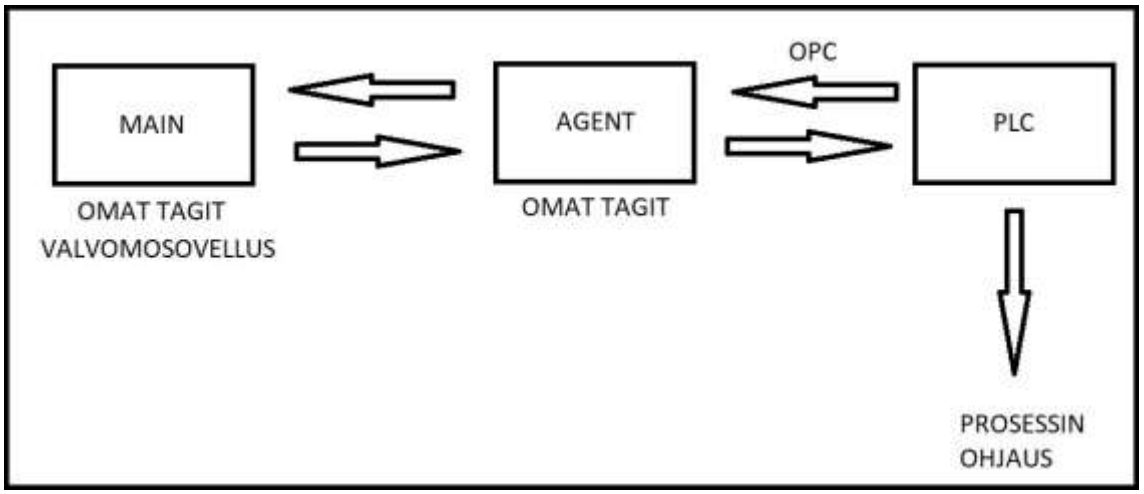


KUVA 12. Pumpun ponnahdusikkuna

Objektit pyrittiin ryhmittelemään niin, että samaa asiaa koskevat objektit ovat lähellä toisiaan, jotta ponnahdusikkunan käyttö olisi operaattorille loogista ja helppoa. Objektien sijoittelussa ja sommittelussa pyrittiin samaan kuin pääikkunoita tehtäessä eli siihen, että objektien välille jäisi tarpeeksi tyhjää tilaa, ja siihen, että ne olisivat samassa linjassa niin pysty- kuin vaakasuunnassakin.

5.2 Tagien luonti

Jätevedenpuhdistamo sisältää pääkoneen, jota kutsuttiin mainiksi, sekä yhden alakoneen, jota kutsuttiin agentiksi. Työssäni olin siis tekemisissä pääkoneen sekä yhden alakoneen kanssa. Pääkone pitää sisällään itse valvomosovelluksen sekä omat taginsa. Alakone puolestaan ei pidä sisällään minkäänlaista sovellusta, vaan pelkät omat taginsa, jotka saavat arvonsa logiikalta OPC-tiedonsiirron standardin avulla. Toisin sanoen alakone toimii siis pääkoneen ja logiikan välillä tiedonkerääjänä sekä sen välittäjänä. Tästä esimerkki alla olevassa kuvassa 13. Koska pääkoneella sekä alakoneella on omat taginsa, niin samat tagit täytyi tehdä kahteen kertaan hieman erilaisilla asetuksilla ja ominaisuuksilla varustettuna molemmille koneille.



KUVA 13. Jätevedenpuhdistamon tietokoneet ja PLC

Työni tärkein osio valvomon valmistumisen kannalta oli varmasti tagien luonti ja varsinkin niiden kansiopolun rakentaminen. Myöhemmin tässä opinnäytetyössä tullaan huomaamaan, että tagien kansiopolku ja sen hyväksi käyttäminen on suuressa roolissa valvomon toimimisen kannalta.

5.2.1 Pää- ja alakoneen tagilistan muokkaaminen

Tagien luominen yksitellen Ignitionissa tapahtuu “Tag Editor”-ikkunassa (ks. kuva 4). Tagien tekeminen yksi kerrallaan olisi ollut todella työlästä, koska tageja piti luoda noin 3700 kappaletta, mutta Ignitionissa tagit voi tehdä myös muulla tavalla. Tagit tuotiin käyttöliittymään kerralla käyttäen Ignitionissa olevaa “Import”- toimintoa, joka siis tuo kaikki tagit joko xml- tai csv-formaattisesta tiedostosta. Tässä työssä päädyttiin käyttämään xml-formaattia, koska csv-formaatti ei toiminut oikein. Ensiksi kuitenkin täytyi muokata tagilistat sekä pää- että alakoneelle Excelissä, jotta ne voitiin myöhemmin muuttaa oikeaan formaattiin, ja lopuksi vielä tuoda valvomoon.

Ensimmäisenä työvaiheena tagien luomisessa oli alkuperäisen tagilistan muokkaaminen alakoneelle sopivaksi. Alkuperäisen tagilistan toimitti jätevedenpuhdistamon logiikkaohjelman suunnittelija (Mipro Oy) ja se oli csv-formaatissa, jolloin muokkaaminen onnistui kätevästi Excelissä. Tagien muokkaaminen aloitettiin sillä, että Excelin taulukkoon luotiin sarakkeet jokaiselle tagin ominaisuudelle,

mitä valvomoon haluttiin tuoda. Kuvassa 14 näkyy esimerkkiä tagien ominaisuuksista Exceliin taulukoituna. Tärkeimpiä tagien ominaisuuksia, jotka täytyi ottaa huomioon tagilistoja, muokatessa olivat tagin nimi, kansiopolku, datatyyppi, OPC-polku, OPC-palvelimen nimi, yksikkö, minimi- ja maksimiarvo ja mahdollinen hälytysominaisuus.

Name	Path	DataType	OPCItemPath
Hairio	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	6	[plc_teravaniemi]DB201,X0.0
Simu_On_Off	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	6	[plc_teravaniemi]DB201,X0.1
AlarajaVaroitus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	6	[plc_teravaniemi]DB201,X0.2
YlarajaVaroitus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	6	[plc_teravaniemi]DB201,X0.3
AlarajaHalytys	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	6	[plc_teravaniemi]DB201,X0.4
YlarajaHalytys	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	6	[plc_teravaniemi]DB201,X0.5
HalytysViive_Asetus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	2	[plc_teravaniemi]DB201,I2.0
SimuAika_Asetus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	2	[plc_teravaniemi]DB201,I4.0
SimuAika	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	2	[plc_teravaniemi]DB201,I6.0
Mittaus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	[plc_teravaniemi]DB201,REAL8.0
Mittaus_Todellinen	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	[plc_teravaniemi]DB201,REAL12.0
Hystereesi	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	[plc_teravaniemi]DB201,REAL16.0
SimuArvo_Asetus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	[plc_teravaniemi]DB201,REAL20.0
AlarajaVaroitus_Asetus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	[plc_teravaniemi]DB201,REAL24.0
YlarajaVaroitus_Asetus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	[plc_teravaniemi]DB201,REAL28.0
AlarajaHalytys_Asetus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	[plc_teravaniemi]DB201,REAL32.0
Alaraja_Vapautusraja	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	[plc_teravaniemi]DB201,REAL36.0
YlarajaHalytys_Asetus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	[plc_teravaniemi]DB201,REAL40.0
Ylaraja_Vapautusraja	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	[plc_teravaniemi]DB201,REAL44.0

KUVA 14. Alakoneen tagilista

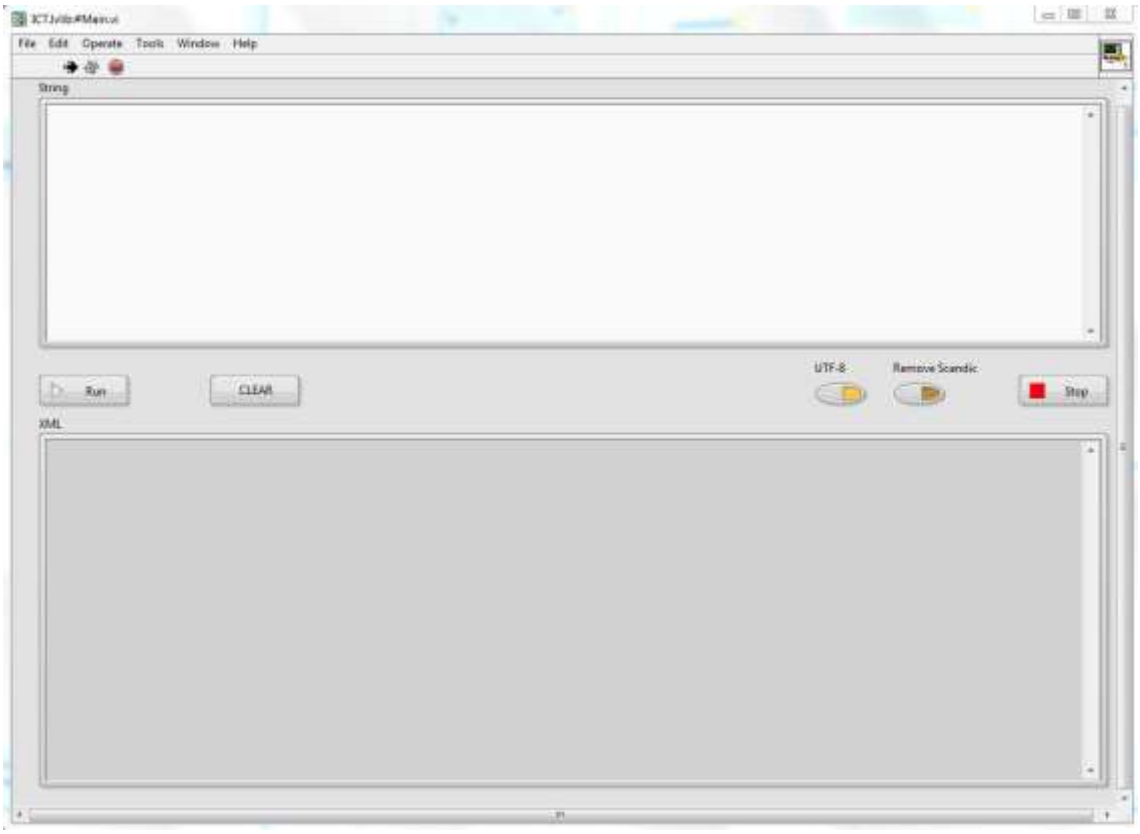
Pääkoneelle tagilista saatiin muokkaamalla alakoneen tagilistaa. Ainoa ero koneiden tagilistojen välillä on OPC-polku eli kuvassa 14 oleva sarake nimeltään ”OPCItemPath”. Alakoneella tagien OPC-polku viittaa suoran logiikan osoitteeseen, kun taas pääkoneella tagit saavat arvonsa alakoneelta. Tämän eroavaisuuden takia pääkoneen tagien OPC-polku ominaisuuteen täytyi laittaa alakoneen tagien kansiopolku logiikan osoitteen sijasta. Esimerkki pääkoneen tagilistasta on alla olevassa kuvassa 15.

Name	Path	DataType	OPCItemPath
Hairio	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	6	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/Hairio
Simu_On_Off	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	6	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/Simu_On_Off
AlarajaVaroitus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	6	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/AlarajaVaroitus
YlarajaVaroitus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	6	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/YlarajaVaroitus
AlarajaHalytys	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	6	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/AlarajaHalytys
YlarajaHalytys	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	6	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/YlarajaHalytys
HalytysViive_Asetus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	2	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/HalytysViive_Asetus
SimuAika_Asetus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	2	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/SimuAika_Asetus
SimuAika	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	2	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/SimuAika
Mittaus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/Mittaus
Mittaus_Todellinen	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/Mittaus_Todellinen
Hystereesi	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/Hystereesi
SimuArvo_Asetus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/SimuArvo_Asetus
AlarajaVaroitus_Asetus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/AlarajaVaroitus_Asetus
YlarajaVaroitus_Asetus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/YlarajaVaroitus_Asetus
AlarajaHalytys_Asetus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/AlarajaHalytys_Asetus
Alaraja_Vapautusraja	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/Alaraja_Vapautusraja
YlarajaHalytys_Asetus	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/YlarajaHalytys_Asetus
Ylaraja_Vapautusraja	Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301	4	ns=2;s=[stptags]Ake/Vesi/Jatevesi/Puhdistamo/Teravaniemi/Valipumppaamo/Mittaukset/FI301/Ylaraja_Vapautusraja

KUVA 15. Pääkoneen tagilista

5.2.2 Tagien tuonti sovellukseen

Tagilistojen valmistuttua voitiin tageja alkaa muuttamaan Excelin csv-formaatista xml-formaattiin. Tämä muutos tehtiin LabVIEW'lla tehtyä ohjelmaa hyväksi käyttäen. Käytännössä ohjelma toimii niin, että Exceliin tehdystä tagilistasta kopioitiin halutut tagit ja liitettiin ne LabVIEW-ohjelmaan (kuva 16). Tämän jälkeen painettiin kuvassa olevaa "Run"-nappia, joka tekee Excelistä kopioidusta tekstistä xml-formaattisen. Kuvassa 17 on esimerkkiä xml-formaatista.



KUVA 16. LabVIEW-ohjelma

KUVA 17. Tagin ominaisuudet xml-formaatissa

Kun pää- ja alakoneen tagilistat oli saatu ajettua LabVIEW-ohjelman läpi, niin tagit voitiin tuoda Ignitioniin. Tämä onnistui siis käyttäen sovelluksen omaa “Import”-toimintoa, jossa valittiin koneelta aikaisemmin tehdyt xml-tiedostot. Tagit tulivat sovelluksen siihen kansioon, mikä tagille oli tiedostossa ominaisuuteen nimeltä “Path” määritetty (ks. kuva 15). Tämä tarkoitti sitä, että kansiot piti luoda Ignitionissa manuaalisesti ennen tagien tuomista sovellukseen.

5.2.3 Tagien ominaisuuksien viimeistely pää- ja alakoneelle

Pää- sekä alakoneen tageihin tuli vielä niiden tuomisen jälkeen muutoksia, joten tagien ominaisuuksia täytyi vielä muokata Ignitionin sisällä. Viimeisiä muutoksia tagien ominaisuuksiin olivat OPC-palvelimen nimi, skannausluokka, historiatiedon keruu ja hälytykset. Muutosten tekeminen tehtiin käyttäen Ignitionin tagiselaimessa olevaa “Tag Search”- työkalua. Tällä työkalulla voi etsiä tageja sovelluksesta eri hakukriteerien mukaan ja muokata löydettyjen tagien ominaisuuksia. Työkalu on siis oivallinen monen tagin yhtäaikaiseen muokkaamiseen.

Yksi muutettavista ominaisuuksista oli OPC-palvelimen nimi (ks. luku 2.2.3). Jokaiselle tagille valitaan OPC-palvelin, joka on määritetty Ignition Gatewayllä. Pääkoneella oleviin tageihin täytyi muuttaa OPC-palvelimen nimeksi “cromiagent012”, joka viittaa jätevedenpuhdistamon alakoneeseen. Kun tämä

muutos oli tehty, niin vasta tällöin pääkoneen tagit pystyivät saamaan arvonsa alakoneen tageista. Alakoneella tagien OPC-palvelimen nimi on vakiosti Gatewayllä määritetty “Cromi OPC-UA Server”.

Toinen muutettavista ominaisuuksista oli skannausluokka (ks. luku 2.2.5). Pääkoneelle tähän ominaisuuteen tuli tilaajan jo valmiiksi Gatewaylle määrittämä skannausluokka nimeltä “cromiagent012”, joka taas viittaa alakoneeseen. Alakoneen skannausluokka pysyi vakiona eli Ignitionin omana skannausluokkana nimeltä “Default”.

Kolmas ja viimeinen muutettava asia oli hälytys- sekä historiankeruu ominaisuuksien muuttaminen (ks. luvut 2.2.4 ja 2.2.5). Tagien hälytykset oli liitetty tageihin jo tagilistojen valmistus vaiheessa. Tästä syystä kaikissa tageissa, joista tulee hälytys valvomoon, oli tämä ominaisuus tehty jo silloin, kun tagit ensimmäisen kerran tuotiin sovellukseen. Myös tagien historiankeruu liitäntä tehtiin erikseen Ignitionin sisällä heti tagien tuomisen jälkeen.

Hälytys- ja historiankeruuominaisuuksien oltua liitettynä tageihin selvisikin, että pääkoneen tageissa pitää olla pelkästään hälytykset päällä ja alakoneella pelkästään historiankeruu. Tämä vaadittava muutos saatiin tehtyä helposti Ignitionin “Tag Search”- työkalulla, jossa pystyy etsimään tageja, joissa historiankeruu- tai hälytysominaisuus on päällä. Näin ollen pääkoneella etsittiin kaikki tagit, joissa historiankeruu oli päällä ja niistä poistettiin tämä ominaisuus. Alakoneella taas etsittiin hälytysominaisuuden omaavia tageja, joista kyseinen ominaisuus poistettiin.

5.3 Tagien liittäminen valvomosovellukseen

Kun kaikki tagit oli saatu tuotua sekä muokattua valvomoon, niin seuraavana työvaiheena oli tagien sijoittelu ikkunoiden objekteihin. Tageja on liitetty objekteihin käyttäen suoraa ja epäsuoraa menetelmää.

5.3.1 Epäsuora menetelmä

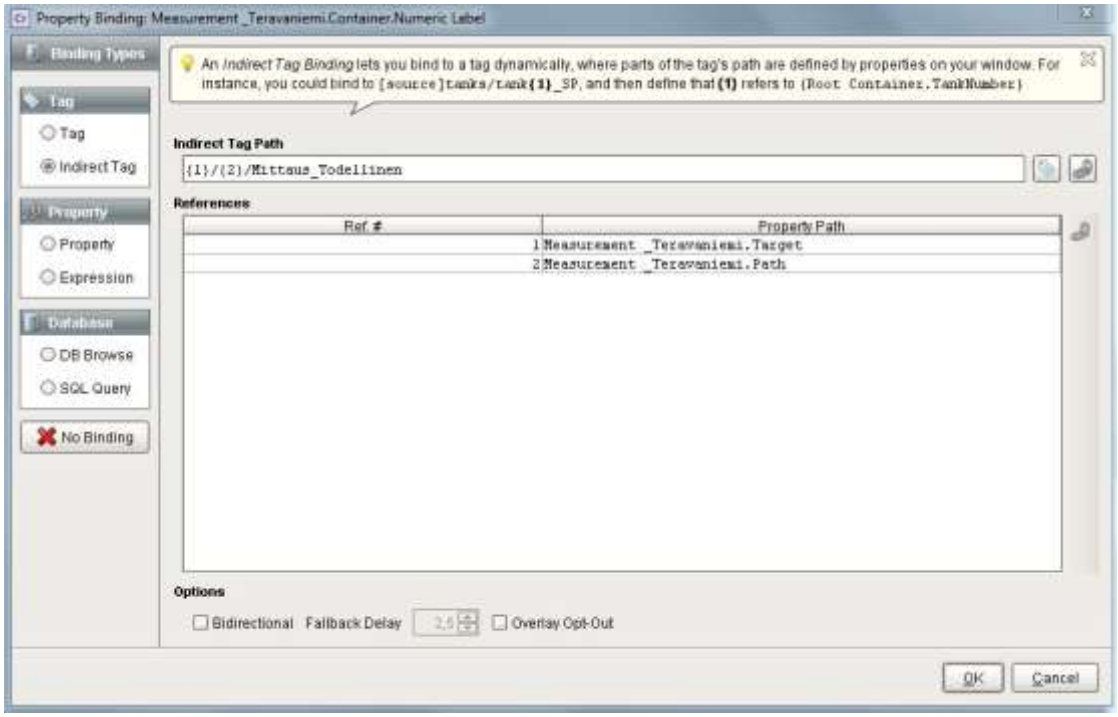
Epäsuorassa tagien liitännöissä haetaan haluttu tagi objektiin kansiopolkua hyväksi käyttäen. Jotta objekteihin saadaan liitettyä oikeat tagit, on objekteissa käytetty hyväksi mukautettuja ominaisuuksia. Nämä ominaisuudet on tarkoitettu muokattavaksi jokaiselle objektille, jonka ikkunalle tuo. Objekteja tuodaan sovelluksen objektipohjien kirjastosta (ks. luku 2.2.6). Mukautetut ominaisuudet olivat pääosin jo valmiiksi luotuna objekteissa, koska työssäni kopioin objektien pohjat vanhasta sovelluksesta. Joissain tapauksissa jouduin tekemään lisää mukautettuja ominaisuuksia objekteihin, jotta tarvittavat toiminnot saatiin suoritettua.

Tärkeimpänä mukautettuna ominaisuutena mainitsen “Target”- ja “Path”-ominaisuudet eli kohde- ja polkuominaisuudet. Molemmat ominaisuudet viittaavat tagin kansiopolkuun ja ovat tyypiltään “string” eli tekstimuotoisia. Kohdeominaisuus kertoo tagin kansiopolun alkuosan ja polkuominaisuus loppuosan. Käytännössä kohdeominaisuus tässä työssä pysyy jokaisella objektilla vakiona, koska kaikki tätä työtä koskevat tagit sijaitsevat kansiorakenteessa samojen pääkansioiden alla. Kohdeominaisuus on siis aina muotoa “Äke/Vesi/Jätevesi/Puhdistamo/Teräväniemi”.

Polku-ominaisuus taas vaihtelee objektien välillä. Tämä ominaisuus kertoo objektille tagin kansiopolun loppuosan eli se on jatkoa kohdeominaisuuden polulle. Se siis määrittelee viimeiset kansiot, mistä objektiin liitettävää tagia luetaan. Polkuominaisuuden kansiopolku vaihtelee sen mukaan, mitä osaprosessia tai laitetta tagi koskee. Polkuominaisuus voi olla esimerkiksi muotoa “Ilmastus/Venttiilit/VC301”, joka siis kertoo, että kyseinen tagi koskee osaprosessia ilmastus sekä se liittyy venttiiliin, jonka positiotunnus on “VC301”.

Valvomon pääikkunoiden objekteihin tagit on lisätty käyttäen epäsuoraa menetelmää. Kun objektille on määritetty oikeat kohde- sekä polkuominaisuudet, niin lopullinen määritys sille, mitä tagia luetaan tai kirjoitetaan, tapahtuu kirjoittamalla tagin nimi erotettuna “/”-merkillä polun loppuun. Alla olevassa kuvassa 18 on esimerkki pääikkunan mittausobjektin epäsuorasta tagi liitännästä. Kuvassa tagia

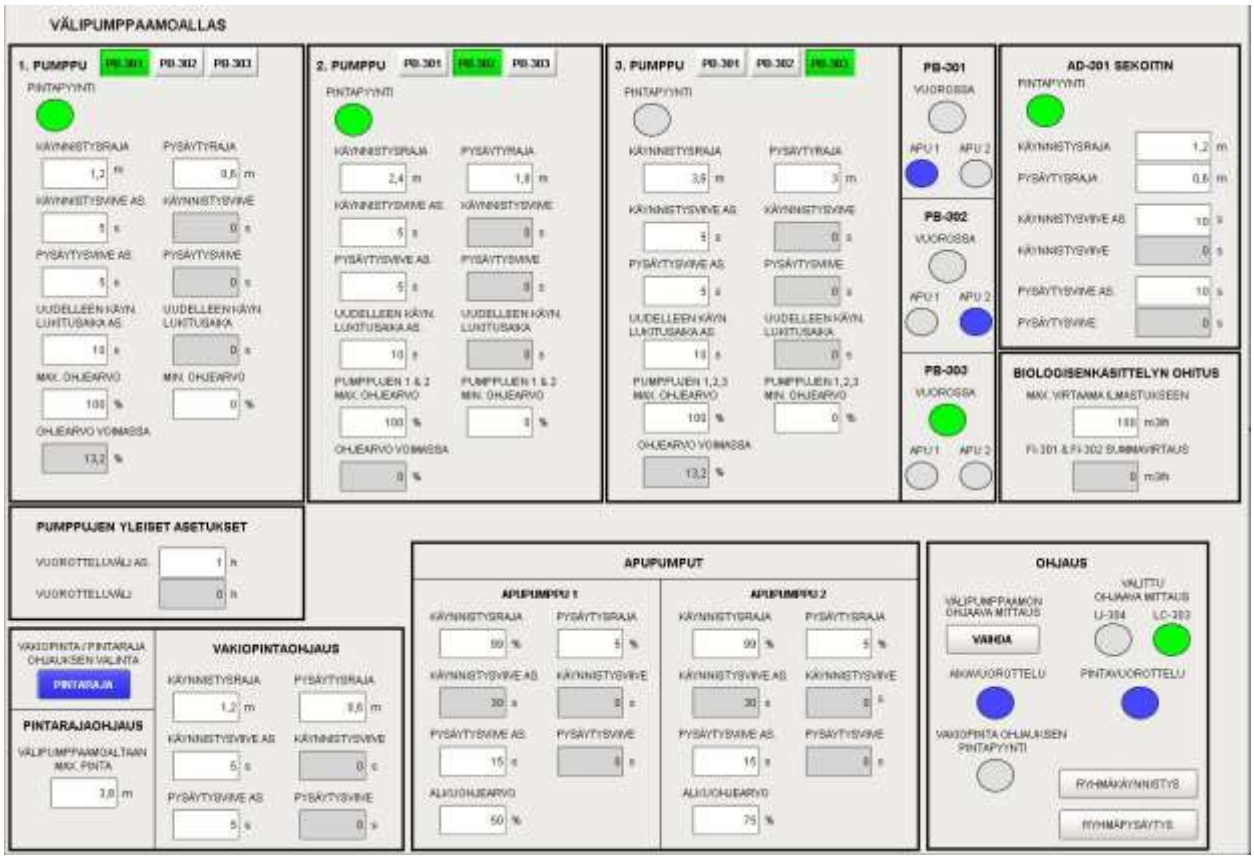
liitetään numeeriseen tekstikenttään, jotta tagin arvoa voidaan seurata. Kuten kuvasta 18 voi huomata, objektipohjan kohde- ja polkuominaisuuksista tehdään referenssejä, joita käytetään hyväksi polun määrittämiseksi.



KUVA 18. Tagin arvon haku objektiin epäsuorasti

5.3.2 Suora menetelmä

Myös prosessissa olevilla altailla, ilmastuksen kompressoreilla, jälkiselkeytyksen jakokaivolla ja ilmastusaltaiden säätimillä on omat ponnahdusikkunansa, jotka aukeavat pääikkunoissa olevista napeista (ks. liite 5). Näissä ponnahdusikkunoissa luetaan muun muassa kohteen hälytyksiä, tiloja ja asetusarvoja sekä ohjataan kohdetta. Alla olevassa kuvassa 19 on esimerkki välipumppaamoaltaan ponnahdusikkunasta.



KUVA 19. Välipumppaamoaltaan ponnahdusikkuna

Näiden ponnahdusikkunoiden objektit on lisätty Ignitionin omasta työkaluvalikosta (ks. kuva 3). Tagien liittäminen ikkunan objekteihin on tehty käyttäen suoraa menetelmää. Tässä menetelmässä ei tarvitse ottaa huomioon tagien kansioita, vaan tagit lisätään objektien ominaisuuksiin suoraan kaikkien tehtyjen tagien joukosta valitsemalla.

5.3.3 Prosessilaitteiden ponnahdusikkunat

Pääikkunoiden prosessilaitteista klikkaamalla avautuvien ponnahdusikkunoiden objektit eivät ole objektipohjien kirjastosta, vaan ne on tehty suoraan Ignitionin omilla työkaluilla (ks. kuva 3.) Tästä syystä aiemmassa luvussa 4.3.1 mainitsemani kohde- sekä polkuominaisuudet ei ole määritelty itse objektissa. Tämän sijasta nämä ominaisuudet on luotu ponnahdusikkunan omiin mukautettuihin ominaisuuksiin, jonka ansiosta yhdellä ponnahdusikkunalla voidaan lukea juuri oikean laitteen tageja.

Kun pääikkunassa olevaa prosessilaitetta klikataan, niin avautuu klikattua laitetta vastaava ponnahdusikkuna. Klikattava prosessilaite on tuotu ikkunaan objekti kirjastosta eli se sisältää kohde- ja polkuominaisuudet. Jotta ponnahdusikkunassa luettaisiin oikeita tageja, niin täytyy nämä ominaisuudet saada siirrettyä ponnahdusikkunan vastaaviin ominaisuuksiin. Alla olevassa kuvassa 20 on koodi, jolla tämä toimenpide suoritetaan. Koodi suoritetaan aina, kun objektia klikataan. Koodin kaksi päätehtävää on avata prosessilaitetta vastaava ponnahdusikkuna ja viedä klikatun objektin kohde- ja polkuominaisuudet ponnahdusikkunan kohde- ja polkuominaisuuksiin. Oikeiden ominaisuuksien oltua ponnahdusikkunassa on niitä käytetty ikkunan objekteissa epäsuoraa menetelmää hyväksi käyttäen (ks. kuva 18).

```
param2 = event.source.parent.Path
param3 = event.source.parent.Target
param4 = event.source.parent.MeasurementIdentifier
param5 = event.source.parent.MeasurementText
popup = unicode(event.source.parent.PopupPath)
system.nav.openWindowInstance(popup, {'Agent' : param1, 'Path' : param2, 'Target'
```

KUVA 20. Prosessilaitetta klikattaessa suoritettava koodi

5.4 Valvomon toiminnot

Käyttöliittymä pitää sisällään välttämättömien nappien, indikointivalojen ja tekstikenttien lisäksi myös muita toimintoja. Nämä toiminnot ovat erillisiä kokonaisuuksia, ja ne vaativat muutakin kuin tagien lukemista tai niihin kirjoittamista. Näiden lisätoimintojen valmistamiseksi joutui perehtymään Ignitionin sisäisiin toimintoihin ja ominaisuuksiin sekä tietokannan kanssa toimimiseen.

Valvomon erillisten toimintojen tehtävä on kertoa prosessissa vallitsevista tiloista tarkemmin sekä helpottaa käyttöliittymän seuraamisessa ja käytössä. Toiminnot, joihin tässä työssä perehdyttiin, olivat trendit eli kaaviot, hälytykset ja kortisto. Nämä toiminnot olivat jo pääosin tehty vanhassa ohjelmassa, mutta niissä oli silti jonkin verran muokkaamista, jotta ne toimisivat minun tekemäni sovelluksen ja tagien kanssa. Tässä luvussa kerron näiden toimintojen pääperiaatteet ja tekemiäni muutoksia niihin.

5.4.1 Hälytykset

Ignitionin hälytykset toimivat tagien avulla niin, että tageihin liitetään erikseen hälytysominaisuus (ks. luku 2.2.4). Kun haluttuihin tageihin on laitettu hälytysominaisuus päälle, sitä on helppo käyttää hyödyksi ikkunoiden objekteissa.

Asia, jossa tageihin laitetut hälytykset näkyvät valvomossa ovat pääikkunoiden objektit. Nämä objektit ovat siis kirjastosta tuotuja, joka tarkoittaa sitä, että niihin on tehty mukautettuja ominaisuuksia, jotta aktiiviset hälytykset saadaan näkymään objekteissa. Valvomon jokaisella laitteella on omat kansiot, jossa laitetta koskevat tagit sijaitsevat. Tämän ansiosta laitekohtaiset hälytykset voidaan lukea laitteen omasta tagikansiosta, ja näin ollen voidaan näyttää laitteen aktiiviset hälytykset värin vaihtona objekteissa.

Objektikirjastoon on objekteille luotu mukautettu ominaisuus nimeltä “Alarm”, joka on datatyypiltään dataset. Dataset-tyyppinen ominaisuus taulukoi saamansa arvot. Tätä ominaisuutta käytetään hyväksi, jotta aktiiviset hälytykset saadaan taulukkoon, jonka sarakkeita ovat tapahtuman tunnusnumero, hälytyksen lähde, hälytystagin kansiopolku, tapahtuma-aika, tila ja hälytyksen prioriteetti.

Alarm-ominaisuus hakee hälytykset taulukkoon käyttäen hyväksi Ignitionin toimintoa nimeltä kiinnitysfunktio. Funktio, jota käytetään, on nimeltään “Alarm Status” eli hälytyksen tila. Tämä funktio etsii tageja, joissa hälytys on päällä ja laittaa ne taulukkoon, jos hälytys on aktiivinen. Funktion hyvä ominaisuus on se, että aktiivisten hälytysten hakualuetta voi rajata kansiopolun avulla, jonka ansiosta laitekohtaisia hälytyksiä voidaan etsiä niiden omista tagikansioista. Tähän rajaukseen on käytetty luvussa 4.3.1 mainitsemaani “Target” - ja “Path”-ominaisuutta, joka siis kertoo kohteena olevan laitteen tagien kansionpolun.

Jokaisen prosessilaitteen symboli pohjassa on myös määritelty mukautettu ominaisuus nimeltään “Status” eli tila. Ominaisuuden arvo muuttuu, jos aiemmin mainitsemassani alarm-ominaisuuden taulukossa on aktiivisia hälytyksiä. Status-ominaisuuden muuttuessa vaihtuu pääikkunassa myös prosessilaitteen väri. Yleisesti valvomossa hälytyksen indikointivärinä on käytetty punaista.

5.4.2 Ponnahdusikkunoiden kaaviot

Kaikissa prosessilaitteista avautuvista ponnahdusikkunoissa on kaavio (ks. kuva 12), joka kertoo valitun laitteen reaaliaikaisia- ja historia-arvoja. Kaavion päällä olevasta napista käyttäjä voi muuttaa kaavion tilaa joko reaaliaikaiseksi tai historialliseksi. Ponnahdusikkunoiden kaaviot on rakennettu vanhan ohjelman kaavioita kopioimalla ja uudelleen muokkaamalla.

Kaikki historiaominaisuuden sisältävät tagit (ks. luku 2.2.5) tallentuvat tietokantaan ja näin ollen vain näiden tagien historia-arvoja voidaan näyttää kaaviossa. Työssä on käytössä MySQL-tietokanta, johon kaikki tagien historiatiedot tallentuvat. Tilaaja on konfiguroinut tietokannan Ignition Gatewaylle nimellä “mysqlmaster”.

Käyttäjän avatessa ponnahdusikkuna tai vaihtaessa kaavion tilaa alkaa kaavio selata tietokantaa, jotta arvoja saadaan kaavioon. Arvojen haku tietokannasta on tehty SQL-komennolla. Komennossa käytetään toimintoa “SELECT”, joka valitsee tietokannan taulukosta arvoja määritettyjen hakukriteerien mukaan. Alla olevassa kuvassa 21 on esimerkki ”SELECT”-toiminnon hakukriteereistä.

```
concat("SELECT 'Mittaus' as Name, concat((SELECT concat('[/',name, ':', provider, ']')
FROM sqlth_drv where id = (SELECT drvid FROM sqlth_scinfo
WHERE id = (SELECT scid FROM sqlth_te where tagpath like '",{Root Container.Target},"/",{Root Container.Path},"/Mittaus_Todellinen' and retired is NULL)))
```

KUVA 21. SQL-komennon hakukriteerit

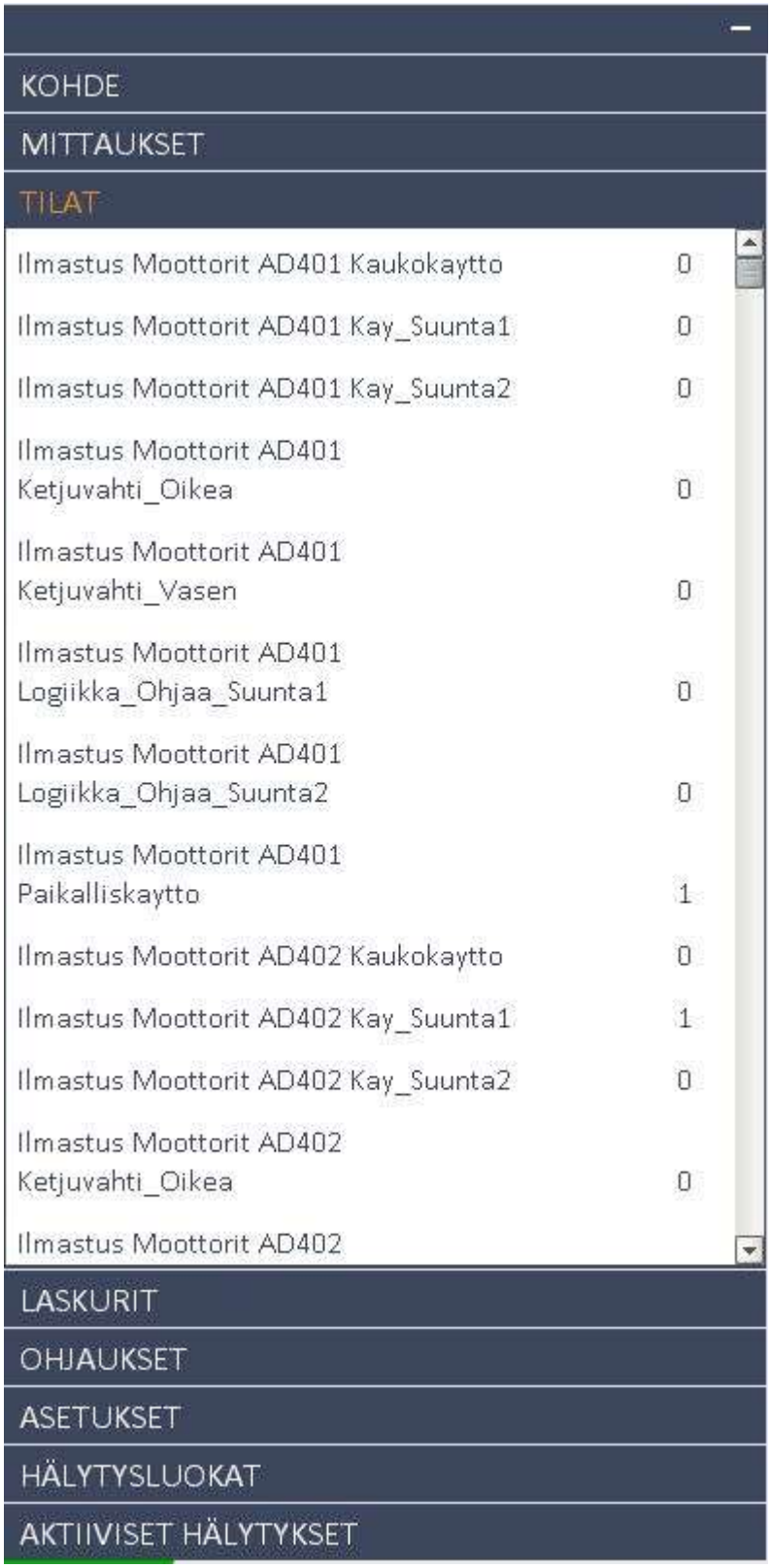
Jotta kuvan 21 komennolla löydettäisiin juuri oikean laitteen historiatietoja, käytetään taas hyväksi tagien kansiopolkuominaisuuksia “Target” ja “Path”. Kuten ylläolevasta SQL-komennosta huomaa, arvojen hakua rajataan kansiopolkuominaisuuksien lisäksi myös tagin nimellä, jotta kaaviossa saadaan näytettyä juuri oikeiden tagien arvoja. Esimerkissä (kuva 21) tagin nimi on ”Mittaus_Todellinen”.

5.4.3 Kortisto

Valvomon kortisto on nimensä mukaisesti kortti, joka aukeaa aina tarvittaessa sitä klikkaamalla. Käyttöliittymässä kortistoja sijaitsee pääikkunoilla ja ponnahdusikkunoilla. Tästä syystä kortistopohjia on luotu projektiin kaksi erilaista. Näissä kahdessa eri paikassa kortistojen sisältö on hieman erilainen, mutta ne toimivat samalla periaatteella. Työssäni olen käyttänyt vanhan ohjelman kortistopohjia ja tehnyt niihin tarvittavat muutokset, jotta ne toimisivat tekemieni tagien kanssa.

Pääikkunoiden kortisto kertoo koko jätevedenpuhdistamon prosessilaitteiden mittaukset, tilat, laskurit, ohjaukset, asetukset, hälytysluokat ja aktiiviset hälytykset eri välilehdillä (kuva 22). Ponnahdusikkunan kortisto taas kohdistuu vain avatun laitteen mittauksiin, tiloihin, laskureihin, ohjauksiin, asetuksiin ja hälytysluokkiin.

Tietokannassa on taulukko nimeltä “sqlt_core”, joka tallentaa kaikkien valvomossa olevien tagien määrittelyt ja nykyiset arvot. Tätä taulukkoa on käytetty hyväksi, jotta kortistoihin on saatu tagien reaaliaikaiset arvot. Tällä tietokannan osalla ei ole mitään tekemistä tagien historiatietojen kanssa, vaan tagin määrittelyt ja nykyiset arvot tallentuvat tietokantaan, vaikka niissä ei ole historiankeruuominaisuus päällä (vrt. luku 2.2.5).



KUVA 22. Pääikkunan kortisto

Kortiston sisällä on välilehtiä kaikille erilaisille tageille. Kuten kuvasta 22 voi huomata, esimerkiksi tiloihin ja mittauksiin liittyvät tagit sijaitsevat eri lehdillä. Nämä

välilehdet saavat arvonsa “List”-objektiin eli listaan, joka on tehty Ignitionin omalla työkalulla. Listaan on tehty SQL-komento, jossa käytetään “SELECT”-toimintoa. Tällä tavoin saadaan haettua listaan haluttujen tagien oloarvot.

Esimerkki pääikkunan kortiston SQL-komennosta on kuvassa 23. Kuten kuvasta huomaa, komento etsii tageja pelkästään sieltä, missä tagin kansiopolun alkuosa on ”Target”-ominaisuuden muotoa. “Target”-ominaisuus on tässä työssä aina muotoa “Äke/Vesi/Jätevesi/Puhdistamo/Teräväniemi”. Tämä tarkoittaa siis sitä, että haettavia tageja ovat vain ne, jotka liittyvät Teräväniemen jätevedenpuhdistamoon.

KUVA 23. Esimerkki SQL-komennosta pääikkunan kortistossa

Pelkkä polku ei riitä hakualueen rajaamiseksi, vaan sitä pitää myös rajata tagien nimillä. Kuvasta 23 voi huomata, että kaikki kortiston välilehteen halutut tagit on listattu käyttäen “OR”-operaattoria tagien välillä. Lopulta komento siis etsii kaikkia tageja, joissa kansiopolun alkuosa on muotoa “Äke/Vesi/Jätevesi/Puhdistamo/Teräväniemi”, ja jossa tagin nimi on jokin listatuista tageista.

5.5 FAT-testi

FAT-testi on toiminnallinen testi, jossa käydään läpi tehtyä työtä ja sen toimivuutta kohteessa. Tämän työn FAT-testin tarkoituksena oli testata valvomosovelluksen toimivuus logiikkaohjelman kanssa. Paikalla testissä oli toimittaja eli Protoconilta minä ja esimieheni (Sami Österman), työn tilaaja (Äänekosken Energia Oy, Atte Myllylä) ja logiikkaohjelman toimittaja (Mipro Oy, Lauri Nykänen). Protacon osallistui testiin etänä VPN-yhteydellä, jonka avulla päästiin kiinni valvomoon. Kommunikointi testiin osallistujien kesken tapahtui Google Hangouts -sovelluksen avulla. Testi kesti kaksi päivää.

Logiikkaohjelmaa simuloitiin syöttämällä siihen erilaisia arvoja, jonka avulla valvomon toimivuutta voitiin arvioida. Testissä käytiin läpi suurin osa valvomossa olevista mittauksista, laitteista ja asetuksista. Näistä testattiin muun muassa hälytyksien toimivuus, mittausarvojen näkyminen valvomossa ja asetusarvojen muutokset. Testausta tehtiin molempiin suuntiin, joka tarkoittaa sitä, että logiikkaohjelmasta pakotettiin arvoja ja tarkasteltiin tulevatko ne perille valvomoon ja asetusarvoja sekä ohjauksia tehtiin valvomon päästä ja katsottiin tulevatko ne perille logiikalle. Hälytyksien tiloja simuloitiin tekemällä ne aktiiviseksi logiikkaohjelmassa ja katsomalla ilmestyvätkö hälytyksistä tarvittavat indikoinnit valvomoon.

Testissä käytettiin kahta eri tietokonetta. Toisella katsottiin valvomokuvaa ajotilassa ja toisella pystyttiin tekemään testin aikana ilmenneet korjaukset valvomosovellukseen. Korjaukset saatiin helposti päivitettyä myös ajotilassa olevaan valvomoon, joten tilaajakin pystyi heti näkemään tehdyt muutokset. Enemmän työtä vaativat korjaukset jätettiin testin ulkopuolelle, ja ne tehtiin muutaman päivän sisään valmiiksi heti testin jälkeen. Kaikki testissä käydyt asiat ja korjaukset näkyvät FAT-testin pöytäkirjassa (liite 1).

Testin aikana tulleita korjauksia olivat visuaalisia ja toiminnallisia virheitä. Visuaalisia korjauksia olivat muun muassa ryhmittelyparannuksia, värityksen vaihtamista, sommittelun parantamista, mittausten yksiköiden korjaamista ja muita peruskorjauksia.

Yksi merkittävä toiminnallinen korjaus oli nappien ja numeeristen tekstikenttien korjaaminen niin, että ne lähettävät tietoa logiikalle. Kirjoittaminen logiikalle näillä objekteilla ei onnistunut ensin, koska yksi asetus oli jäänyt laittamatta päälle. Tämä ominaisuus oli tagien kirjoittamiseen vaadittava ”Bidirectional”-ominaisuus. Asetus näkyy kuvan 18 alalaidassa. Tämä asetus ei ensin ollut päällä, joka tarkoitti sitä, että objekti ei voinut kirjoittaa arvoja, vaan se pystyi vain saamaan niitä. Kun asetus laitettiin päälle, objekti muuttui kaksisuuntaiseksi, ja näin ollen se pystyi nyt myös kirjoittamaan arvoja tageihin.

6 POHDINTA

Työni tarkoituksena oli siis suunnitella ja tehdä jätevedenpuhdistamolle uusi valvomosovellusosio vanhaan valvomoprojektiin. Työ saatiin tehtyä aikataulussa ja nyt Äänekosken Teräväniemen jätevedenpuhdistamolla on käytössä uusi valvomosovellus. Uuteen valvomo-osioon sisältyi osaprosessit välipumppaamo, ilmastus ja jälkiselkeytys. Uusi sovellus pitää sisällään uudet ikkunat, tagit, hälytykset, kortistot ja käyrästöt ponnahdusikkunoissa edellä mainituissa osaprosesseissa.

Vanhasta valvomoprojektista, jonka sisään tein uuden osion, oli paljon apua omassa työskentelyssäni. Työni olisi varmasti vienyt enemmän aikaa ja vaivaa, jos vanhaa ohjelmaa ei olisi ollut apuna ja mallina. Joihinkin isoihin kokonaisuuksiin, mitä työhön kuului, ei minun tarvinnut edes tehdä muutoksia, koska ne toimivat jo automaattisesti myös minun tekemieni tagien kanssa. Esimerkiksi erillinen hälytysikkuna (kuva 6) oli luotu projektiin jo valmiiksi, ja siihen tulivat automaattisesti myös minun tekemäni hälytykset.

Osan objekteista kopion vanhasta ja muokkasin niitä tarpeen mukaan. Tämä vaati minulta paljon tutustumista vanhan ohjelman objekteihin ja niiden toimintaperiaatteisiin. Jos vanhan ohjelman objekteja ei voinut järkevästi muokata niin, että ne toimisivat uusien tagien ja sovelluksen kanssa, niin joissain tapauksissa oli helpompi tehdä asiat alusta asti itse.

Ignitionissa tagien nimillä ja kansiorakenteella on todella suuri vaikutus siihen, miten käyttöliittymä toimii, koska niin monesti objektien muuttaminen tapahtuu tagien kansiopolkuun viittaamalla. Tageja luodessani en vielä tiennyt, että niiden nimillä ja kansiopolulla on näin suuri merkitys. Koska tein tagit eri tavalla kuin edellinen käyttöliittymän suunnittelija, tuli minulle hieman lisätöitä, jotta sain liitettyä tekemäni tagit vanhan käyttöliittymän objekteihin. Jos olisin rakentanut tagit samalla tavalla kuin vanhassakin käyttöliittymässä, olisi monelta lisätyöltä vältytty ja valvomosovellus olisi valmistunut nopeammin. Toisaalta lisätyö teki minulle ai-

noastaan hyvää, koska sen avulla opin enemmän Ignition-ohjelmasta ja ylipäätään käyttöliittymän suunnittelusta, kun pääsin perehtymään vanhaan valvomosovellukseen ja tekemään uutta valvomosovellusta.

Mielestäni työ oli tarpeeksi haastava ja mielenkiintoinen. Työni aikana olen oppinut paljon uutta valvomosovelluksen suunnittelusta. Esimerkiksi ennen työn aloittamista en tiennyt tietokannoista ja niiden kanssa toimimisesta juuri mitään, mutta nyt työn jälkeen on tietokannasta ja sen kanssa työskentelystä tullut tutumpaa. Myös Pythonilla koodaamisesta on tullut uutta kokemusta.

Olen päässyt hyvin perille Ignition SCADA-ohjelmasta. Jos Ignitionilla nyt pitäisi alkaa työstämään uuttaa valvomoprojektia aivan alusta, uskoisin, että se onnistuisi paljon helpommin kuin ennen opinnäytetyön tekemistä. Opinnäytetyöni jälkeen olen jatkanut Teräväniemen jätevedenpuhdistamon valvomoprojektin tekemistä seuraavaan vaiheeseen, joka sisältää loput osaprosessit jäteveden puhdistuksesta.

LÄHTEET

Inductive University. 2018a. About Ignition’s Modular Architecture. Video. Viitattu 25.1.2018, https://inductiveuniversity.com/video/about-ignitions-modular-architecture

Inductive University. 2018b. About templates. Video. Viitattu 26.1.2018, https://inductiveuniversity.com/video/about-templates/7.9?r=%2Fvideo%2Fsearch%2F%3Fq%3Dtemplates%26checked_topics%3D

Inductive University. 2018c. Anatomy of a window. Video. Viitattu 25.1.2018, https://inductiveuniversity.com/video/anatomy-of-a-window

Inductive University. 2018d. Component event handler. Video. Viitattu 29.1.2018, https://inductiveuniversity.com/video/component-event-handlers

Inductive University. 2018e. Component overview. Video. Viitattu 25.1.2018, https://inductiveuniversity.com/video/component-overview

Inductive University. 2018f. Configure alarm on a tag. Video. Viitattu 26.1.2018, https://inductiveuniversity.com/video/configure-alarm-on-a-tag

Inductive University. 2018g. Create alarm journal profile. Video. Viitattu 26.1.2018, https://inductiveuniversity.com/video/create-alarm-journal-profile

Inductive University. 2018h. Creating components. Video. Viitattu 25.1.2018, https://inductiveuniversity.com/video/creating-components

Inductive University. 2018i. Creating opc tags manually. Video. Viitattu 26.1.2018, https://inductiveuniversity.com/video/creating-opc-tags-manually

Inductive University. 2018j. Display status of single alarm. Video. Viitattu 26.1.2018, https://inductiveuniversity.com/video/display-status-of-single-alarm

Inductive University. 2018k. Drawing tools overview. Video. Viitattu 25.1.2018, https://inductiveuniversity.com/video/drawing-tools-overview

Inductive University. 2018l. Event object. Video. Viitattu 29.1.2018, https://inductiveuniversity.com/video/event-object

Inductive University. 2018m. Keeping tags organized. Video. Viitattu 26.1.2018, https://inductiveuniversity.com/video/keeping-tags-organized

Inductive University. 2018n. Script builders. Video. Viitattu 29.1.2018, https://inductiveuniversity.com/video/script-builders

Inductive University. 2018o. Scripting in Ignition. Video. Viitattu 29.1.2018, https://inductiveuniversity.com/video/scripting-in-ignition

Inductive University. 2018p. Set up tags to log. Video. Viitattu 26.1.2018, https://inductiveuniversity.com/video/set-up-tags-to-log

Inductive University. 2018q. Standard Architecture. Video. Viitattu 25.1.2018, https://inductiveuniversity.com/video/standard-architecture.

Inductive University. 2018r. Tags in Ignition. Video. Viitattu 25.1.2018, https://inductiveuniversity.com/video/tags-in-ignition

Inductive University. 2018s. Types of tags. Video. Viitattu 26.1.2018, https://inductiveuniversity.com/video/types-of-tags

Inductive University. 2018t. Window types. Video. Viitattu 25.1.2018, https://inductiveuniversity.com/video/window-types

Kabata, L. 2018. Jätevedenpuhdistus. Osa pro gradu työstä. Blogi. Viitattu 25.2.2018, https://jatevedenpuhdistus.wordpress.com/jatevedenpuhdistus/puhdistusprosessi/biologinen-puhdistus/

Koskinen, T. 2017. Yksikköprosessien ohjaustapaselostus. Ohje. Viitattu 25.2.2018

Kotisalo, A. 2007. Scalable vector graphics. Lahden ammattikorkeakoulu. Mediatekniikan koulutusohjelma. Opinnäytetyö. Viitattu 29.1.2018, https://www.theseus.fi/bitstream/handle/10024/11561/2008-02-26-07.pdf?sequence=1&isAllowed=y.

Metsämäki, M. 1995. Näytön graafinen suunnittelu. Viitattu 29.1.2018, https://oiva.oamk.fi/kirjasto/edita/e_kirjat/uusmedia/37-2465-4.pdf

Schneider Electric Software. 2014. Wonderware Situational Awareness. Video. Viitattu 5.2.2018, https://www.youtube.com/watch?time_continue=72&v=Cwux1hqPM0U

tutorialspoint. 2018. Python-Overview. Kurssi. Viitattu 29.1.2018, https://www.tutorialspoint.com/python/python_overview.htm

Vehmas, J. 2012. Paremman käytettävyyden ohjeistus. Protaconin ohje. Viitattu 29.1.2018

Wonderware HMI SCADA. 2014. Situational Awareness Actionable Alarm Management. Video. Viitattu 5.2.2018, https://www.youtube.com/watch?v=-wBOp3cpqAo

Äänekosken Energia Oy. 2018. Vesi. Viitattu 25.2.2018, https://www.aanekoskenenergia.fi/vesi/

Äänekosken kaupunkisanomat. 2017. Äänekosken Energia investoi 9 miljoonaa jätevedenpuhdistamoon. Uutinen. Viitattu 25.2.2018, http://aksa.fi/aanekosken-energia-investoi-9-miljoonaa-jatevedenpuhdistamoon/

LIITTEET

Liite 1 FAT-testin pöytäkirja

Liite 2 Välipumppaamon PI-kaavio

Liite 3 Ilmastuksen PI-kaavio

Liite 4 Jälkiselkeytyksen PI-kaavio

Liite 5 Välipumppaamon ja ilmastuksen valvomoikkuna

Liite 6 Jälkiselkeytyksen valvomoikkuna

Liite 7 Päänäytön valvomoikkuna

Äänekosken Energia Oy

TERÄVÄNIEMEN JÄTEVEDENPUHDISTAMON SANEERAUS JA LAAJENNUS

VAIHEEN 1.

(VÄLIPUMPPAAMO, ILMASTUS, JÄLKISELKEYTYS)

SW FAT TESTAUSPÖYTÄKIRJA

vp16033

DOKUMENTIN TIEDOT JA VERSIOHISTORIA

Tunniste	Laatija	Tarkastanut	Hyväksynyt	Tyyppi
PRO-048641 Rev. 0.3	Lauri Nykänen	.	.	Pöytäkirja
Revisio	Pvm.	Tekijä	Muutos	
0.3	05.01.2018	LN	Kommenteille	
0.4	8.1	Sami Österman	Protcon lisäkset	

SISÄLLYSLUETTELO

1 SW FAT-TESTAUS

1.1 Paikka

- Äänekosken Energian valvomotilassa (Kotakennääntie 31, Äänekoski)

1.2 Ajankohta

- 3. - 4.1.2018

1.3 Osallistujat

Lauri Nykänen	Mipro Oy
Sami Österman	Protacon Oy (etänä/Oulu)
Atte Väyrynen	Protacon Oy (etänä/Oulu)
Atte Myllylä	Äänekosken Energia Oy

1.4 Testausympäristö

- prosessinvalvonnan käyttöliittymä
- automaatiokeskuksen AK1 logiikka (CPU1)

1.5 Testauksen laajuus

Vaihe 1.

- välipumppaamo
- ilmastus
- jälkiselkeytys

1.6 Testausmenetelmä

- tietoliikenneyhteydet logiikan ja valvomon välillä
- I/O-liityntöjen testaus valvomon käyttöliittymää vasten
- mallipiirien testaaminen
- prosessiautomaation sovellusohjelmien testaaminen

1.7 Järjestelmärakenteen ja liikennöinnin testaus

Testataan logiikan ja valvomon välinen tietoliikenneyhteys.

Huomiot:

- todettiin järjestelmän rakenteen ja liikennöinnin vastaavan suunnitelmia.

Tarkastuksen tulos: **HYVÄKSYTTY**

1.8 I/O-liityntöjen testaaminen

1.8.1 Digitaalitulojen testaus (tila- ja hälytystiedot, DI)

Digitaalitulojen testaus toteutetaan pakottamalla jokainen tulo 0 ja 1 tilaan ohjelmointilaitteella. Tulos havaitaan valvomonäytöiltä.

- tarkastetaan, että hälytykset tulevat hälytys- ja tapahtumalistalle
- tarkastetaan hälytysten toimisuunta

Huomiot:

- testaustulos merkittiin projektikohtaiseen testauskansioon

Tarkastuksen tulos: **HYVÄKSYTTY**

1.8.2 Digitaalilähtöjen testaus (ohjauslähdöt, DO)

Digitaalilähtöjen testaus toteutetaan ohjaamalla valvomon käyttöliittymästä jokainen lähtö 0 ja 1 tiloihin. Tulos havaitaan ohjelmointilaitteella.

- testataan ohjattavien kojeiden ristiriitahälytykset

Huomiot:

- testaustulos merkittiin projektikohtaiseen testauskansioon

Tarkastuksen tulos: **HYVÄKSYTTY**

1.8.3 Analogiatulojen testaus (mittaustulot, AI)

Analogiatulojen testaus toteutetaan simuloimalla jokainen tulo ohjelmointilaitteella. Tulos havaitaan valvomonäytöiltä.

- tarkastetaan mittausten skaalaus
- testataan analogiatulojen ylä- ja alarajahälytykset

Huomiot:

- testaustulos merkittiin projektikohtaiseen testauskansioon

Tarkastuksen tulos: **HYVÄKSYTTY**

1.8.4 Analogialähtöjen testaus (ohjearvolähdöt, AO)

Analogialähtöjen testaus toteutetaan asettamalla jokaiseen lähtöön ohjearvo valvomon käyttöliitty-mästä. Tulos havaitaan ohjelmointilaitteella.

Huomiot:

- testaustulos merkittiin projektikohtaiseen testauskansioon

Tarkastuksen tulos: **HYVÄKSYTTY**

1.9 Mallipiirien testaaminen

Testataan mallipiirien toiminta. Testattavat mallipiirit:

- UDT_VACON (taajuusmuuttaja, väylä)
- UDT_SIMOCODE (moottorinohjain, suorakäyttö, väylä)
- UDT_MITTAUS (analogiatulo)
- UDT_MITTAUS_SC (mittaus, väylä)
- UDT_VENTTIILI (venttiili)
- UDT_SAADIN (PID-säädin)
- UDT_NAYTE (näytteenotin)

1.9.1 **UDT_VACON (PB-301)**

Toiminnot:

- asetusarvot binäärinen: SCADA -> PLC ei toimi, korjattu **OK**
- asetusarvot numeerinen: SCADA -> PLC ei toimi, korjattu **OK**
- indikoinnit: **OK**
- mittaukset: **OK**
- yksiköt: **OK**

Huomiot:

Tarkastuksen tulos: **HYVÄKSYTTY**

1.9.2 **UDT_SIMOCODE (AD-301)**

Toiminnot:

- asetusarvot binäärinen: SCADA -> PLC ei toimi, korjattu **OK**
- asetusarvot numeerinen: SCADA -> PLC ei toimi, korjattu **OK**
- indikoinnit: **OK**
- mittaukset: **OK**
- yksiköt: **OK**

Huomiot:

Tarkastuksen tulos: **HYVÄKSYTTY**

1.9.3 **UDT_MITTAUS (LC-303)**

Toiminnot:

- asetusarvot binäärinen: SCADA -> PLC ei toimi, korjattu **OK**
- asetusarvot numeerinen: SCADA -> PLC ei toimi, korjattu **OK**
- indikoinnit: **OK**
- mittaukset: **OK**
- yksiköt: **OK**

Huomiot:

Tarkastuksen tulos: **HYVÄKSYTTY**

1.9.4 **UDT_MITTAUS_SC (QC-402)**

Toiminnot:

- asetusarvot binäärinen: SCADA -> PLC ei toimi, korjattu **OK**
- asetusarvot numeerinen: SCADA -> PLC ei toimi, korjattu **OK**

- indikoinnit: **OK**
- mittaukset: **OK**
- yksiköt: **OK**

Huomiot:

- varoitustieto puuttuu käyttöliittymässä

Tarkastuksen tulos: **HYVÄKSYTTY**

1.9.5 **UDT_VENTTIILI (VC-401)**

Toiminnot:

- asetusarvot binäärinen: SCADA -> PLC ei toimi, korjattu **OK**
- asetusarvot numeerinen: SCADA -> PLC ei toimi, korjattu **OK**
- indikoinnit: **OK**
- mittaukset: **OK**
- yksiköt: **OK**

Huomiot:

Tarkastuksen tulos: **HYVÄKSYTTY**

1.9.6 **UDT_SAADIN (FIC-302)**

Toiminnot:

- asetusarvot binäärinen: SCADA -> PLC ei toimi, korjattu **OK**
- asetusarvot numeerinen: SCADA -> PLC ei toimi, korjattu **OK**
- indikoinnit: **OK**
- mittaukset: **OK**
- yksiköt: **OK**

Huomiot:

Tarkastuksen tulos: **HYVÄKSYTTY**

1.9.7 **UDT_NAYTE (QP-301)**

Toiminnot:

- asetusarvot binäärinen: SCADA -> PLC ei toimi, korjattu **OK**
- asetusarvot numeerinen: SCADA -> PLC ei toimi, korjattu **OK**
- indikoinnit: **OK**
- mittaukset: **OK**
- yksiköt: **OK**

Huomiot:

Tarkastuksen tulos: **HYVÄKSYTTY**

1.10 Prosessiautomaation sovellusohjelmien testaaminen

Sovellusohjelmien toiminnot pohjautuen seuraaviin dokumentteihin:

- yksikköprosessien ohjaustapaselostus, 11.4.2017, Ramboll

1.10.1 **Välipumppaamo**

Toiminnot:

- asetusarvot: **OK**
- laskennalliset arvot: **OK**
- ryhmäkäynnistys/ryhmäpysäytys: **OK**
- ohjaavan pintamittauksen valinta: **OK**
- sekoittimen käynnistys/pysäytys: **OK**
- pumppujen pintarajaohjaus: **OK**
- pumppujen vakiopintaohjaus: **OK**
- biologisen käsittelyn ohitus: **OK**
- säätöpiirit: **OK**

Huomiot:

- jos käyttäjä valitsee saman pumpun useampaan kertaan käyttöön, niin tehdään logiikan sovellusohjelmaan pakotus, että viiveen (esim. 30 s) jälkeen logiikkaa pakottaa pumput järjestykseen 1-2-3, jos käyttäjä ei ole sitä ennen muutosta tehnyt.

Tarkastuksen tulos: **HYVÄKSYTTY**

1.10.2 **Ilmastus**

Toiminnot:

- asetusarvot: **OK**
- laskennalliset arvot: **OK**
- ryhmäkäynnistys/ryhmäpysäytys: **OK**
- lohkon tilan valinta anoksinen/hapellinen: **OK**
- nitraattikierrätys: **OK**
- ilmastuksen tehosekoitus: **OK**
- ylijäämälietteen poisto: **OK**

Huomiot:

- jos käyttäjä valitsee saman kompressorin useampaan kertaan käyttöön, niin tehdään logiikan sovellusohjelmaan pakotus, että viiveen (esim. 30 s) jälkeen logiikkaa pakottaa kompressorit järjestykseen 1-2-3, jos käyttäjä ei ole sitä ennen muutosta tehnyt.
- optimipaine-säädön paineen lisäyksen/vähennyksen viiveiden muuttujien kommenttiin muutettava s –> min.
- Nitraattikierrätyksen pumpuille lisättävä tuottokäyrät käyttöliitttymään.
- TA-402 Ilmastusallas 2, lohko 1 lämpötilamittaus puuttuu käyttöliittymästä.
- TA-403 Ilmastusallas 3, lohko 1 lämpötilamittaus puuttuu käyttöliittymästä.

Tarkastuksen tulos: **HYVÄKSYTTY**

1.10.3 **Jälkiselkeytys**

Toiminnot:

- asetusarvot: **OK**
- laskennalliset arvot: **OK**
- ryhmäkäynnistys/ryhmäpysäytys: **OK**
- palautuslietteen pumppaus: **OK**

- pintalietteen poisto: **OK**

Huomiot:

- jos käyttäjä valitsee saman kompressorin useampaan kertaan käyttöön, niin tehdään logiikan sovellusohjelmaan pakotus, että viiveen (esim. 30 s) jälkeen logiikkaa pakottaa kompressorit järjestykseen 1-2-3, jos käyttäjä ei ole sitä ennen muutosta tehnyt.
- optimipaine-säädön paineen lisäyksen/vähennyksen viiveiden muuttujien kommenttiin muutettava s –> min.
- Nitraattikierrätyksen pumpuille lisättävä tuottokäyrät käyttöliitttymään.

Tarkastuksen tulos: **HYVÄKSYTTY**

2 MUUT ASIAT

2.1 Testissä tehdyt havainnot, selvitettävät asiat ja korjausta/muutosta vaativat asiat

Välipumppaamo:

- jos käyttäjä valitsee saman pumpun useampaan kertaan käyttöön, niin tehdään logiikan sovellusohjelmaan pakotus, että viiveen (esim. 30 s) jälkeen logiikkaa pakottaa pumput järjestykseen 1-2-3, jos käyttäjä ei ole sitä ennen muutosta tehnyt.

Ilmastus:

- jos käyttäjä valitsee saman kompressorin useampaan kertaan käyttöön, niin tehdään logiikan sovellusohjelmaan pakotus, että viiveen (esim. 30 s) jälkeen logiikkaa pakottaa kompressorit järjestykseen 1-2-3, jos käyttäjä ei ole sitä ennen muutosta tehnyt.
- optimipaine-säädön paineen lisäyksen/vähennyksen viiveiden muuttujien kommenttiin muutettava s –> min.
- Nitraattikierrätyksen pumpuille lisättävä tuottokäyrät käyttöliitttymään.
- TA-402 Ilmastusallas 2, lohko 1 lämpötilamittaus puuttuu käyttöliittymästä.
- TA-403 Ilmastusallas 3, lohko 1 lämpötilamittaus puuttuu käyttöliittymästä.

2.2 Protacon Technologies lisäykset

Käyttöliittymä muutokset:

- Tehosekoitin napin väri muutettu Vihreäksi, kun sekoitin valittu päälle olevaksi (tehty jälkikäteen)
- Numeeristen indikaattoreiden väri muutettu harmaamaksi
- ”Käytössä/ei käytössä” -nappi muutettu siten että näkyy vain teksti ”käytössä” tai ”Ei käytössä” valitun tilan mukaan. Väri Vihreä, kun ”käytössä” (jälkikäteen tehty, ei mainintaa testeissä)
- Trendeihin lisätty kuvaajien nimet sekä mahdollisuus muuttaa viivan väriä
- Varoitusindikointi väylämittausten popup ikkunoihin (lisätty jälkikäteen)

3 LIITTEET

Liite 1. Testauskansio

4 SW FAT TESTAUKSEN TULOS

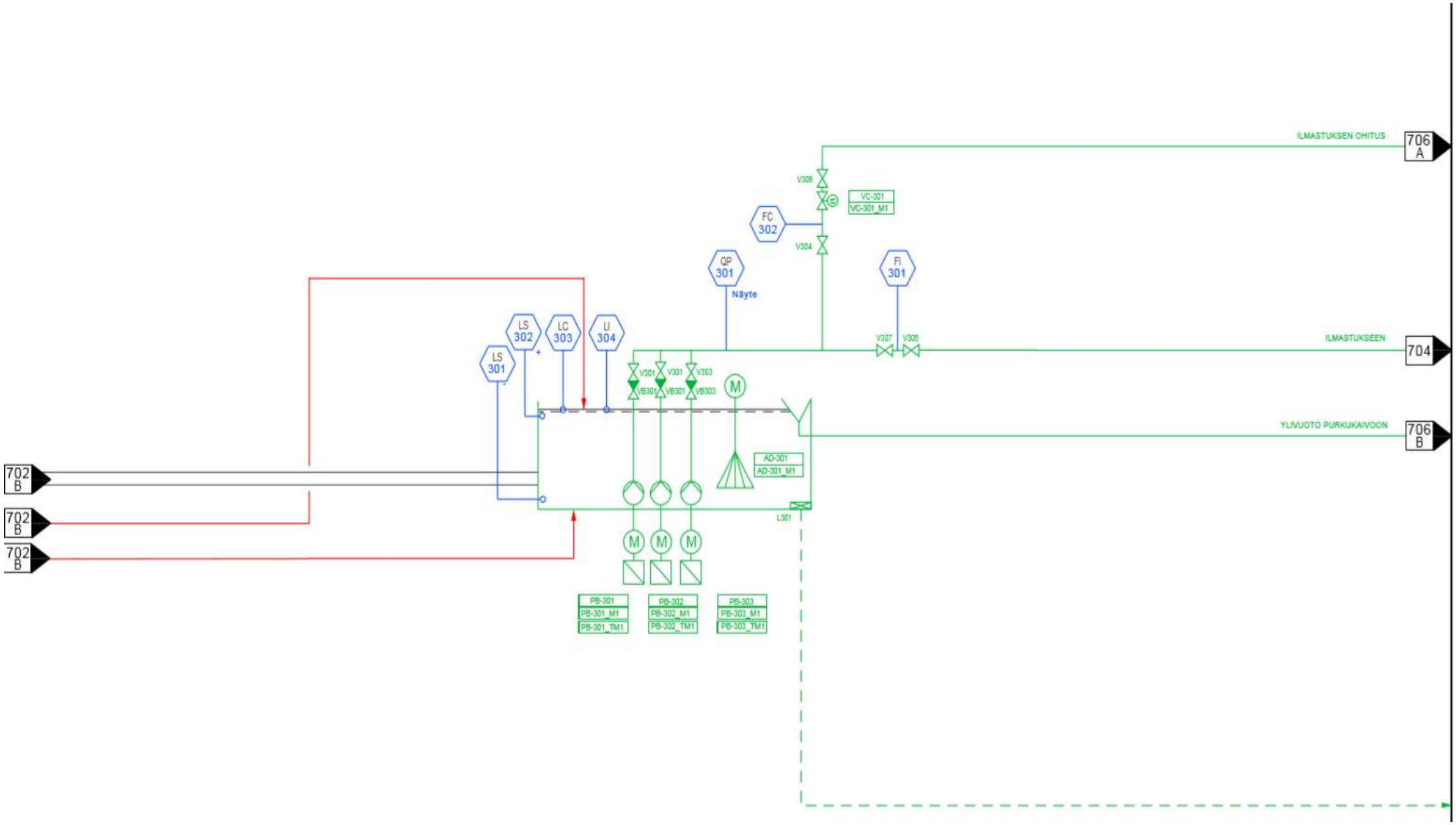
SW FAT testaus: **HYVÄKSYTTY**

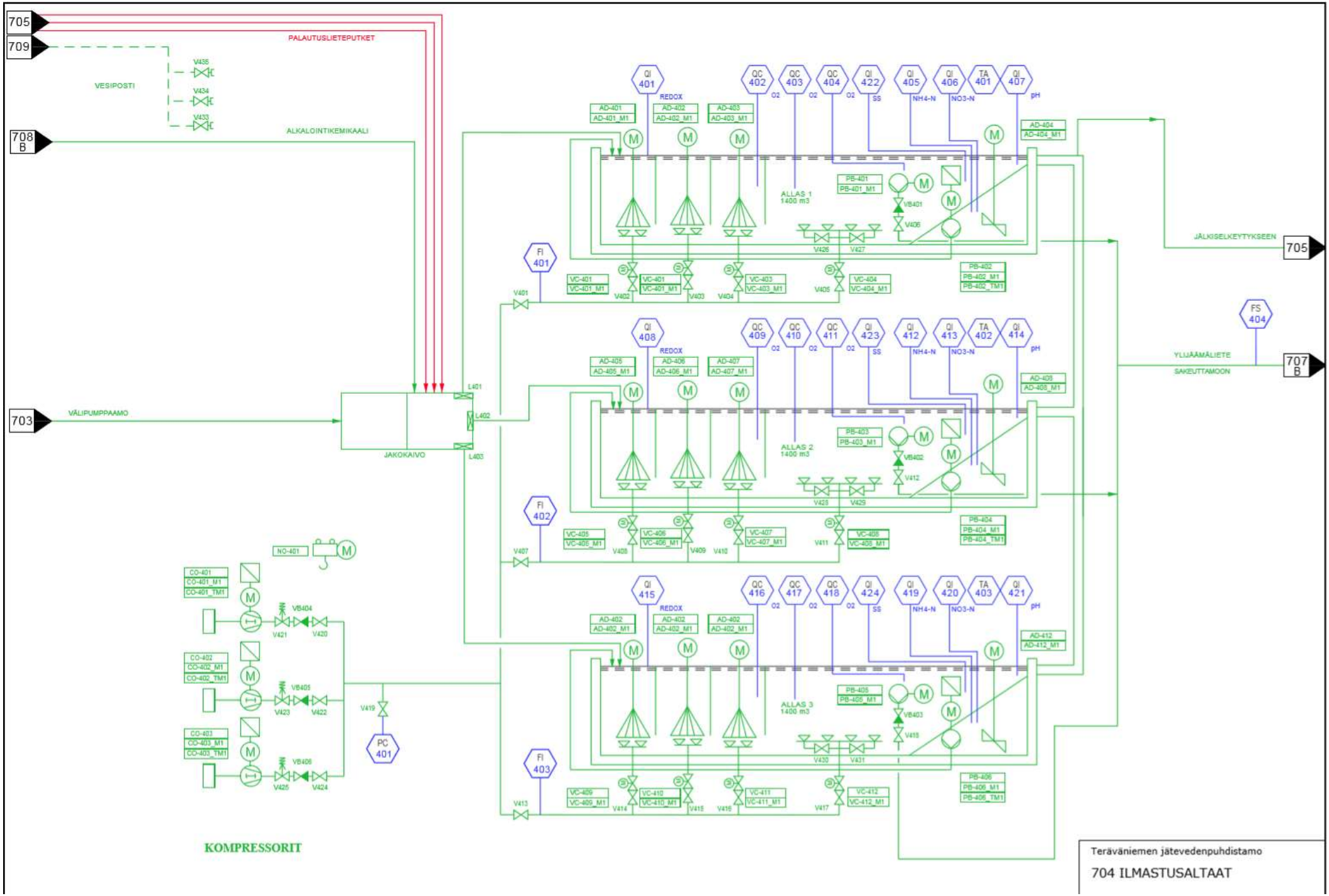
Lauri Nykänen

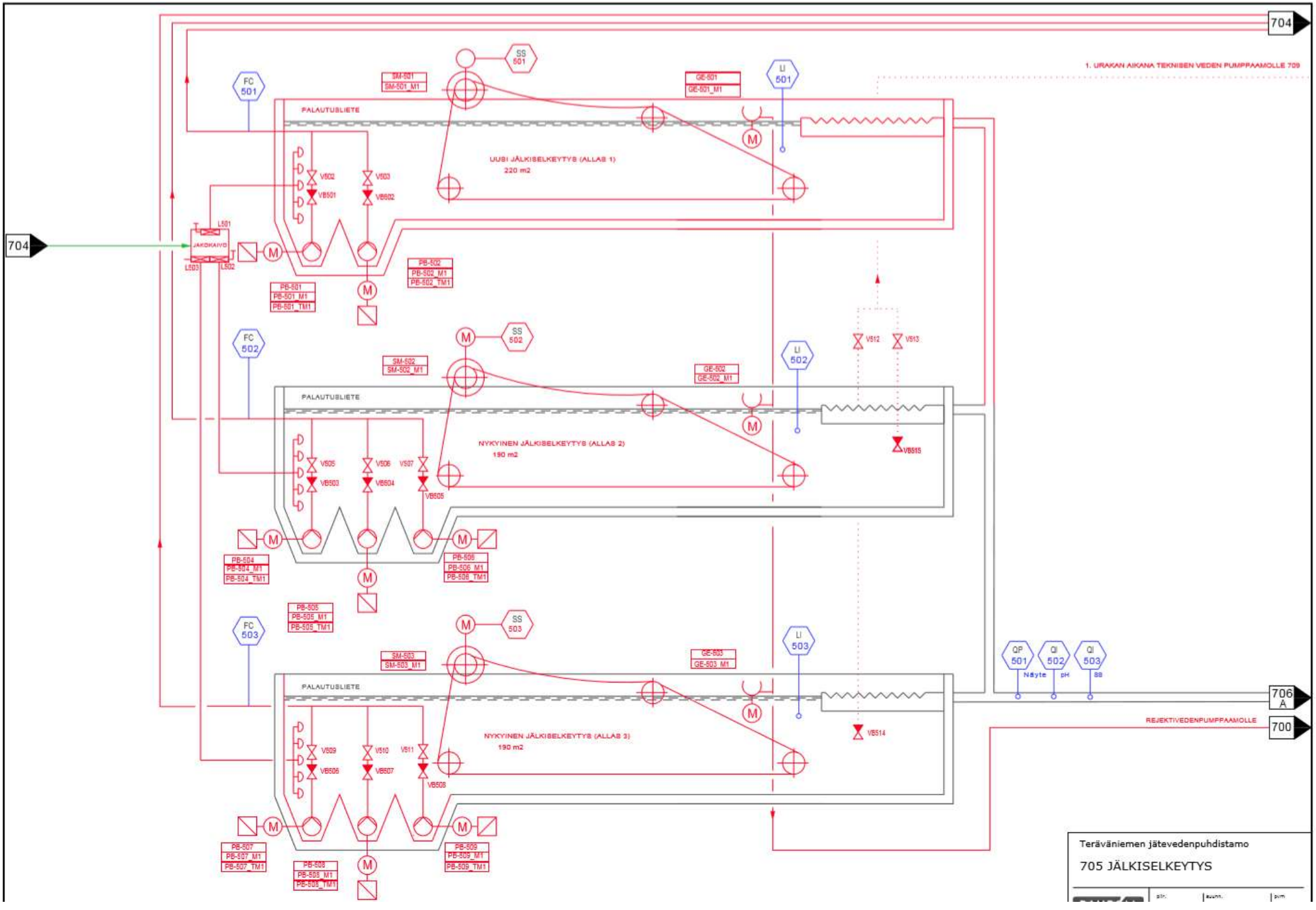
Lead Designer

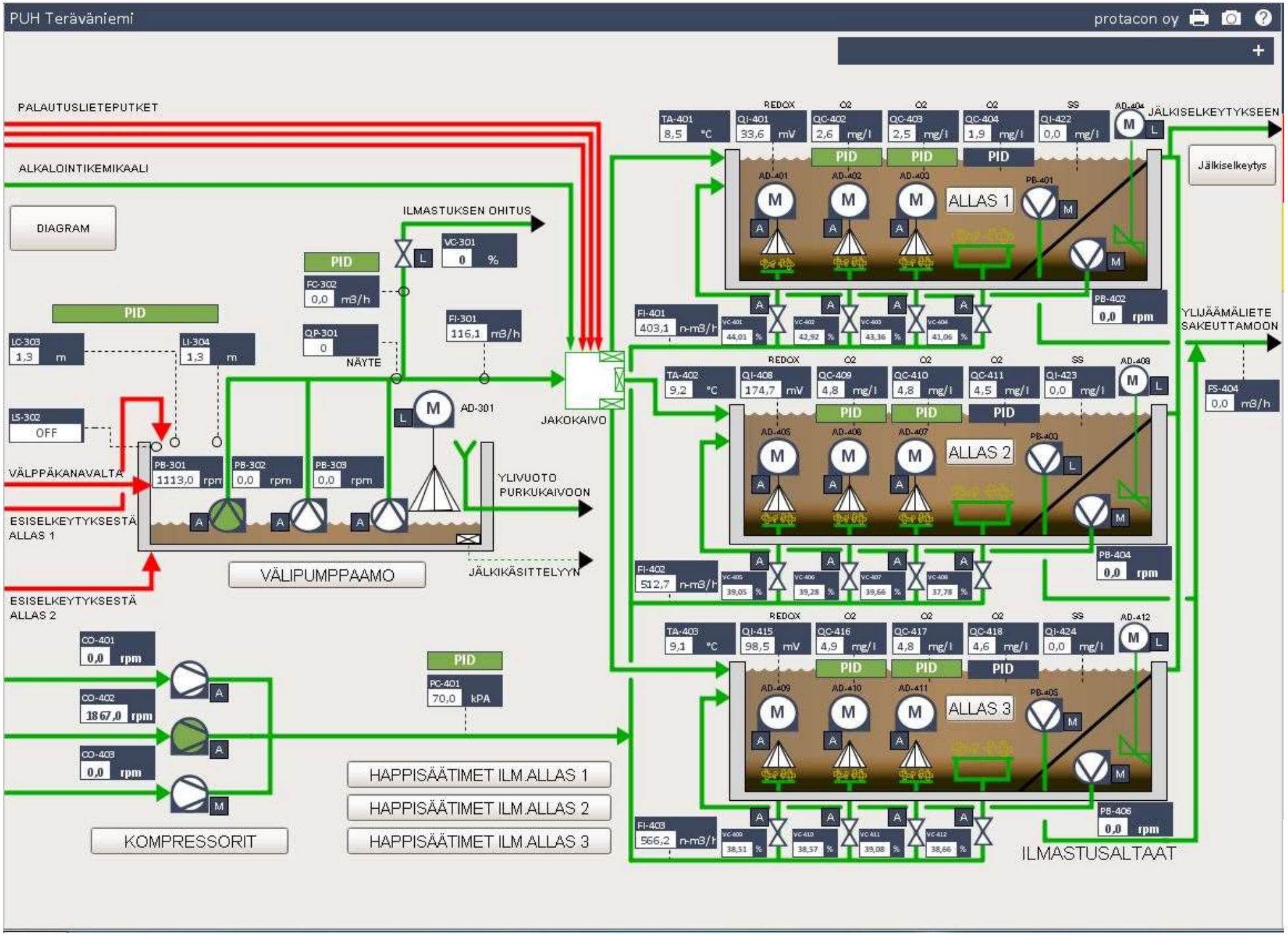
Mipro Oy

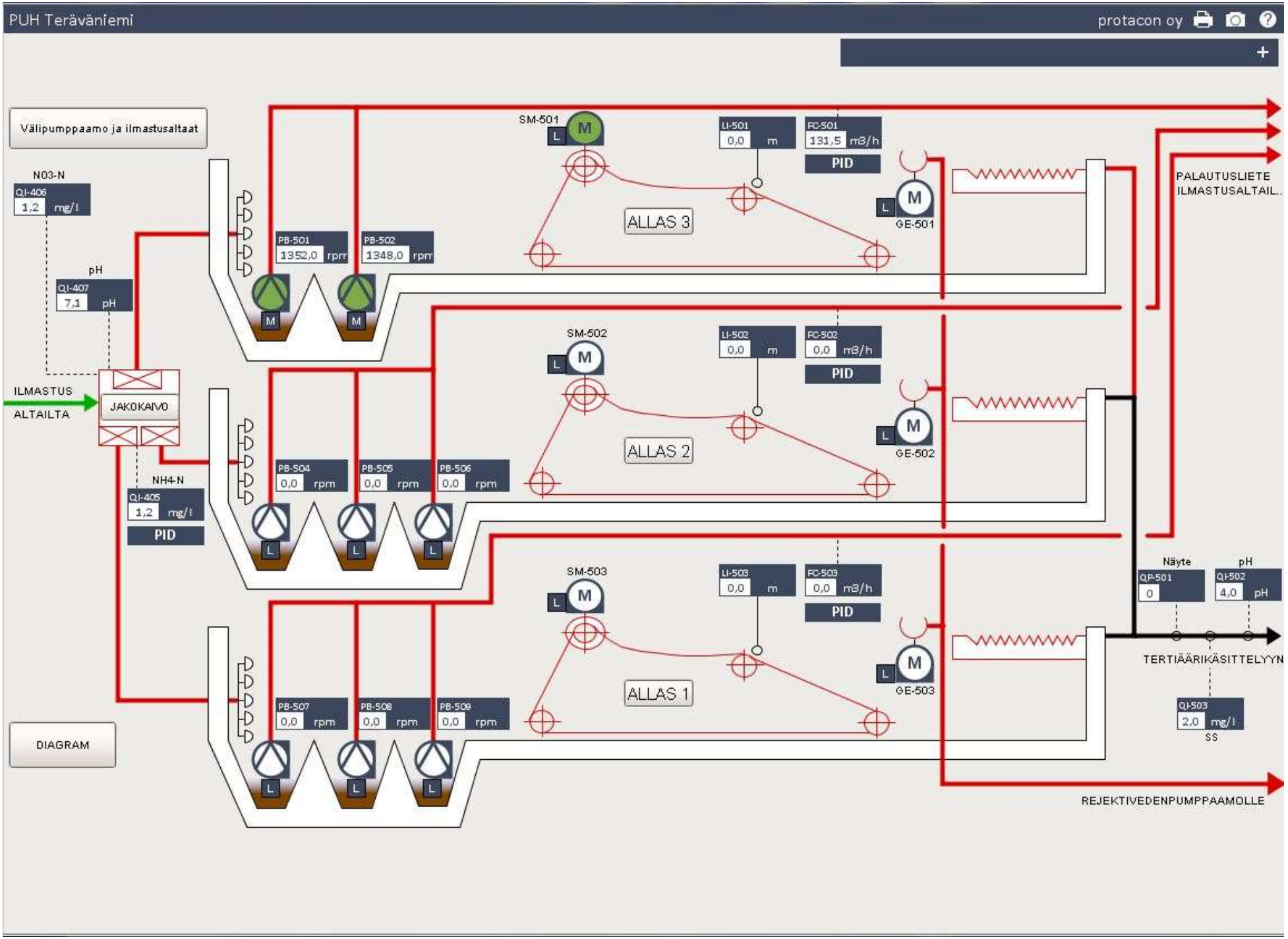
Äänekoskella 4.1.2018

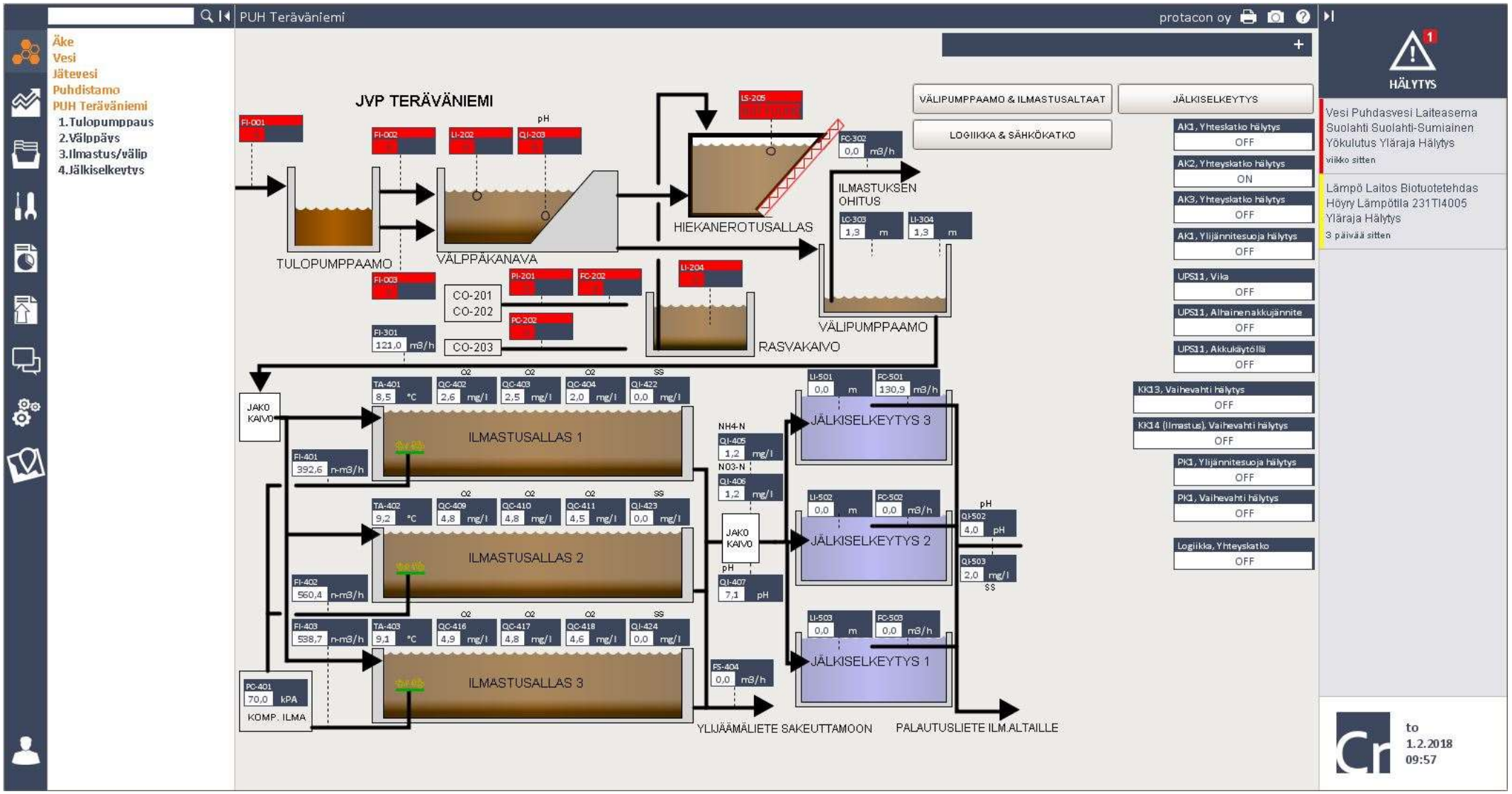

ILMASTUKSEN OHITUS
706 A
V306
VC-301
VC-301_M1
FC 302
V304
QP 301
Näyte
FI 301
LS 302
LC 303
LI 304
LS 301
V307
V305
ILMASTUKSEEN
704
V301
V303
VB301
VB303
M
YLIVUOTO PURKUKAIVOON
706 B
AD-301
AD-301_M1
L301
702 B
702 B
702 B
M
M
M
PB-301
PB-301_M1
PB-301_TM1
PB-302
PB-302_M1
PB-302_TM1
PB-303
PB-303_M1
PB-303_TM1


705
709
PALAUTUSLIETEPUTKET
VESIPOSTI
V435
V434
V433
708 B
ALKALOINTIKEMIKAALI
703
VÄLIPUMPPAAMO
JAKOKAIVO
L401
L402
L403
QI 401
REDOX
QC 402
O2
QC 403
O2
QC 404
O2
QI 422
SS
QI 405
NH4-N
QI 406
NO3-N
TA 401
QI 407
pH
AD-401
AD-401_M1
AD-402
AD-402_M1
AD-403
AD-403_M1
AD-404
AD-404_M1
ALLAS 1
1400 m3
PB-401
PB-401_M1
VB401
V406
V426
V427
FI 401
V401
VC-401
VC-401_M1
V402
V403
V404
V405
PB-402
PB-402_M1
PB-402_TM1
JÄLKISELKEYTYKSEEN
705
FS 404
YLIJÄÄMÄLIETE
SAKEUTTAMOON
707 B
QI 408
REDOX
QC 409
O2
QC 410
O2
QC 411
O2
QI 423
SS
QI 412
NH4-N
QI 413
NO3-N
TA 402
QI 414
pH
AD-405
AD-405_M1
AD-406
AD-406_M1
AD-407
AD-407_M1
AD-408
AD-408_M1
ALLAS 2
1400 m3
PB-403
PB-403_M1
VB402
V412
V428
V429
FI 402
V407
VC-405
VC-405_M1
V408
V409
V410
V411
PB-404
PB-404_M1
PB-404_TM1
QI 415
REDOX
QC 416
O2
QC 417
O2
QC 418
O2
QI 424
SS
QI 419
NH4-N
QI 420
NO3-N
TA 403
QI 421
pH
AD-402
AD-402_M1
AD-412
AD-412_M1
ALLAS 3
1400 m3
PB-405
PB-405_M1
VB403
V415
V430
V431
FI 403
V413
VC-409
VC-409_M1
V414
V415
V416
V417
PB-406
PB-406_M1
PB-406_TM1
NO-401
CO-401
CO-401_M1
CO-401_TM1
VB404
V421
V420
CO-402
CO-402_M1
CO-402_TM1
VB405
V423
V422
CO-403
CO-403_M1
CO-403_TM1
VB406
V425
V424
V419
PC 401
KOMPRESSORIT
Teräväniemen jätevedenpuhdistamo
704 ILMASTUSALTAAT


704
1. URAKAN AIKANA TEKNISEN VEDEN PUMPPAAMOLLE 709
FC 501
SM-501
SM-501_M1
SS 501
GE-501
GE-501_M1
LI 501
PALAUTUSLIETE
UUSI JÄLKISELKEYTYS (ALLAS 1)
220 m2
V502
V503
VB501
VB502
L501
JAKOKAIVO
L503
L502
PB-502
PB-502_M1
PB-502_TM1
PB-501
PB-501_M1
PB-501_TM1
FC 502
SM-502
SM-502_M1
SS 502
GE-502
GE-502_M1
LI 502
V512
V513
PALAUTUSLIETE
NYKYINEN JÄLKISELKEYTYS (ALLAS 2)
190 m2
V505
V506
V507
VB503
VB504
VB505
VB515
PB-504
PB-504_M1
PB-504_TM1
PB-506
PB-506_M1
PB-506_TM1
PB-505
PB-505_M1
PB-505_TM1
FC 503
SM-503
SM-503_M1
SS 503
GE-503
GE-503_M1
LI 503
PALAUTUSLIETE
NYKYINEN JÄLKISELKEYTYS (ALLAS 3)
190 m2
V509
V510
V511
VB506
VB507
VB508
VB514
PB-507
PB-507_M1
PB-507_TM1
PB-508
PB-508_M1
PB-508_TM1
PB-509
PB-509_M1
PB-509_TM1
QP 501
Näyte
QI 502
pH
QI 503
SS
706 A
REJEKTIVEDENPUMPPAAMOLLE
700
Teräväniemen jätevedenpuhdistamo
705 JÄLKISELKEYTYS


PUH Teräväniemi
protacon oy
PALAUTUSLIETEPUTKET
ALKALOINTIKEMIKAALI
DIAGRAM
ILMASTUKSEN OHITUS
VC-301 0 %
PID
FC-302 0,0 m3/h
FI-301 116,1 m3/h
QP-301 0 NÄYTE
PID
LC-303 1,3 m
LI-304 1,3 m
LS-302 OFF
AD-301
JAKOKAIVO
VÄLPPÄKANAVALTA
ESISELKEYTYKSESTÄ ALLAS 1
ESISELKEYTYKSESTÄ ALLAS 2
PB-301 1113,0 rpm
PB-302 0,0 rpm
PB-303 0,0 rpm
YLIVUOTO PURKUKAIVOON
VÄLIPUMPPAAMO
JÄLKIKÄSITTELYYN
CO-401 0,0 rpm
CO-402 1867,0 rpm
CO-403 0,0 rpm
KOMPRESSORIT
PID
PC-401 70,0 kPA
HAPPISÄÄTIMET ILM.ALLAS 1
HAPPISÄÄTIMET ILM.ALLAS 2
HAPPISÄÄTIMET ILM.ALLAS 3
JÄLKISELKEYTYKSEEN
Jälkiselkeytys
YLIJÄÄMÄLIETE SAKEUTTAMOON
FS-404 0,0 m3/h
TA-401 8,5 °C
QI-401 33,6 mV
QC-402 2,6 mg/l
QC-403 2,5 mg/l
QC-404 1,9 mg/l
QI-422 0,0 mg/l
AD-404
ALLAS 1
PB-401
PB-402 0,0 rpm
FI-401 403,1 m3/h
TA-402 9,2 °C
QI-408 174,7 mV
QC-409 4,8 mg/l
QC-410 4,8 mg/l
QC-411 4,5 mg/l
QI-423 0,0 mg/l
AD-408
ALLAS 2
PB-403
PB-404 0,0 rpm
FI-402 512,7 m3/h
TA-403 9,1 °C
QI-415 98,5 mV
QC-416 4,9 mg/l
QC-417 4,8 mg/l
QC-418 4,6 mg/l
QI-424 0,0 mg/l
AD-412
ALLAS 3
PB-405
PB-406 0,0 rpm
FI-403 566,2 m3/h
ILMASTUSALTAAT


PUH Teräväniemi
protacon oy
Välipumppaamo ja ilmastusaltaat
N03-N
QI-406
1,2 mg/l
pH
QI-407
7,1 pH
ILMASTUS
ALTAILTA
JAKOKAIVO
NH4-N
QI-405
1,2 mg/l
PID
PB-501
1352,0 rpm
PB-502
1348,0 rpm
SM-501
LI-501
0,0 m
FC-501
131,5 m3/h
PID
ALLAS 3
GE-501
PB-504
0,0 rpm
PB-505
0,0 rpm
PB-506
0,0 rpm
SM-502
LI-502
0,0 m
FC-502
0,0 m3/h
PID
ALLAS 2
GE-502
PB-507
0,0 rpm
PB-508
0,0 rpm
PB-509
0,0 rpm
SM-503
LI-503
0,0 m
FC-503
0,0 m3/h
PID
ALLAS 1
GE-503
PALAUTUSLIETE
ILMASTUSALTAIL..
Näyte
QP-501
0
pH
QI-502
4,0 pH
TERTIÄÄRIKÄSITTELYYN
QI-503
2,0 mg/l
SS
REJEKTIVEDENPUMPPAAMOLLE
DIAGRAM


PUH Teräväniemi
protacon oy
Äke
Vesi
Jätevesi
Puhdistamo
PUH Teräväniemi
1.Tulopumppaus
2.Välppäys
3.Ilmastus/välip
4.Jälkiselkeytys
JVP TERÄVÄNIEMI
VÄLIPUMPPAAMO & ILMASTUSALTAAT
JÄLKISELKEYTYS
LOGIIKKA & SÄHKÖKATKO
HÄLYTYS
Vesi Puhdasvesi Laiteasema Suolahti Suolahti-Sumiainen Yökulutus Yläraja Hälytys
viikko sitten
Lämpö Laitos Biotuotetehdas Höyry Lämpötila 231TI4005 Yläraja Hälytys
3 päivää sitten
TULOPUMPPAAMO
VÄLPPÄKANAVA
HIEKANEROTUSALLAS
ILMASTUKSEN OHITUS
VÄLIPUMPPAAMO
RASVAKAIVO
ILMASTUSALLAS 1
ILMASTUSALLAS 2
ILMASTUSALLAS 3
JÄLKISELKEYTYS 3
JÄLKISELKEYTYS 2
JÄLKISELKEYTYS 1
JAKO KAIVO
KOMP. ILMA
YLIJÄÄMÄLIETE SAKEUTTAMOON
PALAUTUSLIETE ILM.ALTAILLE
AK1, Yhteskatko hälytys OFF
AK2, Yhteyskatko hälytys ON
AK3, Yhteyskatko hälytys OFF
AK1, Ylijännitesuoja hälytys OFF
UPS11, Vika OFF
UPS11, Alhainen akkujännite OFF
UPS11, Akkukäytöllä OFF
KK13, Vaihevahti hälytys OFF
KK14 (Ilmastus), Vaihevahti hälytys OFF
PK1, Ylijännitesuoja hälytys OFF
PK1, Vaihevahti hälytys OFF
Logiikka, Yhteyskatko OFF
to 1.2.2018 09:57